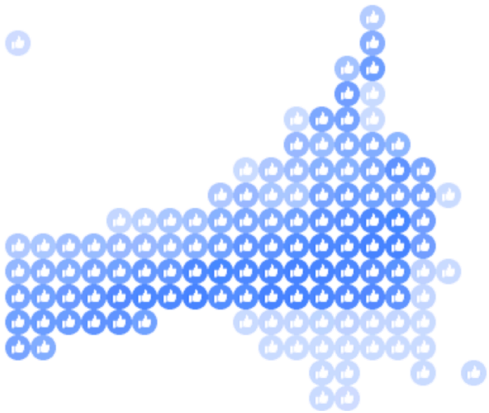
Name: Frederik la Cour
Stud. Nr.: 201707873
Date: 10-08-2018
Characters: 12594/12000

Aesthetic Programming: Reexam



In this project, I have chosen to expand on the weekly mini exercise 4 from the Aesthetic Programming curriculum concerning the topic of data capturing. The main objective of that particular mini exercise was to critically reflect upon the activity of data capturing in digital culture, along with developing a program based on the open call "CAPTURE ALL"[i] from Transmediale 2015. The program would then serve as a starting point for this critical reflection.

In addition to this topic, I further intend to explain in detail how the program, that I have developed for this exam, works and the concept or theme that it addresses. This is achieved by analyzing bits of the code in conjunction with Gerlitz and Helmond's paper: "The like economy: Social buttons and the data-intensive web" (2013)[ii] and Søren Pold's chapter "Button" (2008) from the book: "Software Studies lexicon"[iii].

This will then enable a critical reflection of the activity of data capturing in our digital culture, and then finally discuss the question regarding; how my program is demonstrating the ability to integrate practical skills and theoretical understandings in order to conceptualize, articulate and develop computational artifacts that are able to critically examine or address the aesthetic, cultural, social and political aspects of software. Here, I will be using evidence from the previous sections of this paper.

Section 1: My Program

My program consists of a white canvas on which is drawn a matrix grid made of the well-known Facebook like buttons. At first glance, the buttons may seem to be appearing randomly in the matrix. However, the user may discover that the webcam is engaged, and as a result sees a connection between the like buttons' appearance and the movement of the person in front of the screen. If the user opens the console for this program they discover a set of data rapidly accumulating with data generated from the active like buttons. The data flow is meant to symbolize the way that real-life likes are converted into a commodity in the economy of Facebook, which then, continuing with the symbolism, should have created popup advertisement objects for every 1000 like recorded in the console.log. However, due to the short time span that I had for working

with this software, I unfortunately wasn't able to finish creating the advertisement objects and implement them in the program.

Now that we have established how the program presents itself towards the user, I will now move on to the more technical aspects of how it functions. My program fits into the topic of data capturing because it captures video data via the webcam. The live video data is then further analyzed for RGBA (Red, Green, Blue, Alpha) values (fig. 1), which are processed in a function that extract the brightness values of each individual pixel from the live video data. I then create a matrix of rectangles using nested for-loops. Each of the rectangles represents a pixel of very low resolution live webcam feed.

My initial idea was to convert the matrix of rectangles into a matrix of DOM element buttons, however due to the lack of properties in regards to having more than one state I was unable to use actual DOM buttons in my matrix. I then tried using checkboxes which have two states, either checked or unchecked. However, after numerous failed attempts of styling them as like buttons through CSS I finally settled on using the image-function in JS/p5.js and imported a .png of a like button. I discovered that, by having used an image, I could utilize the tint function combined with the gathered RGBA data in order to make each image display the individual brightness values for each pixel. The amount of tint for each like button is determined in a logical statement that proclaims that if the brightness value is greater than the threshold value, then the pixel will be displayed. If not, the pixel will *not* be displayed.

The console.log is activated every time a pixel is in the active state, this will trigger a message consisting of a "$" followed by "x+bright" and the word "Like" (fig. 2). For every 1000 likes an Advert object was supposed to appear in order to reinforce the concept of like commodification. Unfortunately, that particular function was not able to be completed due to lack of time. Nonetheless, I did manage to create the Advert object in pseudo code. It was supposed to be rectangles of various sizes and positions that would have gradually appeared.

Section 2: The examination of code and conceptual conjunctions

As previously explained, my program is centered on capturing video data from a webcam and analyzing it in order to extract the RGBA values. This data is then used to create a matrix system using the nested for-loops. The idea here is that creating this matrix corresponds to symbolically creating a framework for social interaction on the web in the same fashion that other social media services have, such as Facebooks social button interaction system. Feeding data into this particular matrix is essentially an interaction just like pressing a button with your mouse. Although, it is slightly different.

```
video.loadPixels();
for (var y = 0; y < video.height; y++) {
  for (var x = 0; x < video.width; x++) {
       var index = (x + y * video.width)*4;
         var r = video.pixels[index+0]; //  Red
         var g = video.pixels[index+1]; //  Green
         var b = video.pixels[index+2]; //  Blue
         var a = video.pixels[index+3]; //  Alpha
```

Fig. 1: RGBA

Name: Frederik la Cour
Stud. Nr.: 201707873
Date: 10-08-2018
Characters: 12594/12000

Normally when we encounter buttons we tend to have a conceptual model of how to perform an interaction with it and what that interaction results in. Pold writes: “A button indicates a functional control; something well defined and predictable will happen as a result of the user pressing it”[iv]. This definition only applies to how mechanical buttons works. In software, buttons consist of a set of visual abstractions which enables a more arbitrary user approach towards its function. In truth, the visual buttons in a piece of software are there to resemble mechanical buttons and aid user interaction with the interface.[v] The software conception of a button is quite apparent in the way the user is able to interact through my program.

Rather than giving the user the feeling of a singular controllable button that has an effect that corresponds to a classic mechanical comprehension of how buttons work, I programmed the buttons in such a way that the user is constantly interacting with them by just being present in front of the webcam. Thus, the result is a continuous and constant interaction with the multiple buttons in the matrix. This takes away one of the important purposes of the button, which is forcing the user into a process of binary decision making[vi]. Instead, when the user is interacting with multiple buttons simultaneously it creates a sensation of uncontrollability, resulting in an unconscious action of engaging with the software.

```
if (bright > threshold) {
              tint(255,bright)
              console.log('$',x+bright,'Like');
              //number of likes contributing to the Like economy
      } else {
              tint(255,0)
      }
```

Fig. 2: console.log()

Another important part of the software is the conditional statement, in which each like in the matrix is assigned its individual value. Because each pixel in the matrix is actually an image and not a button, it creates this interesting dynamic, where there is a gradual range in the expression of likes. If the conditional statement was controlling an actual button it would control the binary properties to either be active or inactive. However, because we are utilizing the RGBA data that pulls floating point values between 0.00 and 255.00, and not binary integers (1 or 0). It enables the visual abstractions to display gratified transparency for each like in the matrix. Resulting in a far more expressive and organic interaction. Using non-binary values as a way of interacting is establishing a connection of human expression, and by displaying the like buttons in different degrees of transparency, it gives the user visual feedback, and makes the interaction with the individual pixels seem more organic due to the seemingly different velocities, just like a human is able to press a button in either a hard or soft manner.

What the user can’t see is how each interaction no matter if it is a seemingly hard or soft interaction is all recorded and stored through the `console.log('$',x+bright,'Like');` line. All of the pixels and all the video data that you are creating are a part of another system running in the backend of the program. This is representing how, as Gerlitz and Helmond put it, "(…) Facebook's

Like economy contributes to the making of an alternative fabric, organised through data flows in the back end."[vii] So while the user is interacting with the software it is collecting large amounts of behavioral data from the user and putting that data up for auction without the user knowing. This is where each bit of data is commodified and sold to the highest bidder, which then enables companies to fund and produce targeted advertisement aimed at the user.[viii]

All of the examples I have analyzed in this section serve to enhance the users' comprehension of my program in terms of understanding how it works in relation to the aspects of our software culture.

Section 3: Discussion/conclusion

In my experience working with different kinds of data capturing I have come across numerous ways in which this concept is executed. Using hardware in the form of light dependent resistors in various .js framework in order to capture data from UV light, or capturing the impendence in the waves of audio recordings and using that as an output for controlling other parameters in certain abstractions. Data capturing is definitely a powerful tool as well as materiel. Capturing outside phenomena and translating it into data, which can be used as an output to control other things, creates a lot of possibilities. However, the use of data capturing in online environments are often happening without the user's knowledge. I find this highly disturbing that someone or something may be collecting behavioral data from your online presence without you having a saying in what this data is used for.

I think that it is important to help create awareness about these things happening in our public domain software environments,. I believe the best way of doing this is by creating critical code or software art that exposes and breaks the "functional spell"[ix] so that it becomes apparent to everyone what is actually happening in the backend of the websites they are using every day.

In relation to how my program demonstrates the ability to integrate practical skills and theoretical understandings in order to conceptualize, articulate and develop computational artifacts, for examining the aesthetic, cultural, social and political aspects of software, I have used my extensive knowledge and skill involving shaping and creating my digital artifact through a thoughtful design process. In this process I carefully evaluated each abstraction that I coded in order to create awareness of the topic of data capturing behavioral data from unknowing users on websites with enabled social media content. This data is then commodified and used in a back-end infrastructure where advertisement is targeted at the user. Thus, by critically examining this issue and using software as a way of articulating, this is achieved through writing the crucial algorithms and assigning conceptual understandings to them; essentially having the code come to represent and articulate the issue itself.

Name: Frederik la Cour
Stud. Nr.: 201707873
Date: 10-08-2018
Characters: 12594/12000

Sources:

'Call for Works 2015 | Transmediale'. Archive. Call for Works 2015. Accessed 13 August 2018. https://transmediale.de/content/call-for-works-2015.

Gerlitz, Carolin, and Anne Helmond. 'The like Economy: Social Buttons and the Data-Intensive Web'. *New Media & Society* 15, no. 8 (December 2013): 1348–65. https://doi.org/10.1177/1461444812472322.

Pold, Søren. 'Button'. In *Software Studies: A Lexicon*, edited by Matthew Fuller, 31–36. Leonardo Books. Cambridge, Mass: MIT Press, 2008.

https://github.com/L4COUR/ApE_Aesthetic_programming-REEXAM/tree/master/p5/ApE_11082018_REEXAM_webcamPixels

https://cdn.rawgit.com/L4COUR/ApE_Aesthetic_programming-REEXAM/0c237e9f/p5/ApE_11082018_REEXAM_webcamPixels/index.html

---

[i] 'Call for Works 2015 | Transmediale', archive, Call for Works 2015, accessed 13 August 2018, https://transmediale.de/content/call-for-works-2015.

[ii] Carolin Gerlitz and Anne Helmond, 'The like Economy: Social Buttons and the Data-Intensive Web', *New Media & Society* 15, no. 8 (December 2013): 1348–65, https://doi.org/10.1177/1461444812472322.

[iii] Søren Pold, 'Button', in *Software Studies: A Lexicon*, ed. Matthew Fuller, Leonardo Books (Cambridge, Mass: MIT Press, 2008), 31–36.

[iv] Søren Pold, 'Button', in *Software Studies: A Lexicon*, ed. Matthew Fuller, Leonardo Books (Cambridge, Mass: MIT Press, 2008), 31.

[v] Søren Pold, 'Button', in *Software Studies: A Lexicon*, ed. Matthew Fuller, Leonardo Books (Cambridge, Mass: MIT Press, 2008), 31–36.

[vi] Pold, 'Button'.

[vii] Carolin Gerlitz and Anne Helmond, 'The like Economy: Social Buttons and the Data-Intensive Web', *New Media & Society* 15, no. 8 (December 2013): 1348–65, https://doi.org/10.1177/1461444812472322.

[viii] Gerlitz and Helmond.

[ix] Pold, 'Button'.