D2.3 Documentation of Prototypes

Jens Ewald (ESR1)



# Abstract

The following documentation provides an account of the development process of interactive prototypes that explored how creative technologist practice can enable nourishing relationships between internet connected technologies and the experience of our bodies. There are three prototypes documented: [Making a Step Tracker](#making-a-step-tracker), [Exploring Data as Ink](#exploring-data-as-ink), and [Making a Drawing Thing](#making-a-drawing-thing). This document firstly captures notes during the making phases, reflections on code and hardware that was integral to design decisions, and secondly documents aspects of making within the co-design process of the final prototype when deployed with a participant.

# Acknowledgements

This work has been supervised by Jayne Wallace (UNN), Mel Woods (UoD), Mehan Jayasuriya (Mozilla Foundation), and Max von Grafenstein (UdK).

This project is part of [Open Design of Trusted Things (OpenDoTT) doctoral training network](https://opendott.org.) and has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 813508.



# Table of Contents

[Abstract 1](#_Toc117859549)

[Acknowledgements 1](#_Toc117859550)

[Table of Contents 2](#_Toc117859551)

[Introduction 3](#_Toc117859552)

[Why focus on step data 4](#_Toc117859553)

[Learning from Open Technology training 5](#_Toc117859554)

[Making a Step Tracker 6](#_Toc117859555)

[Using an online tutorial 7](#_Toc117859556)

[Sensors with built-in step detection 8](#_Toc117859557)

[Finding third party code 11](#_Toc117859558)

[Insights from Making a Step Tracker 12](#_Toc117859559)

[Exploring Data as Ink 13](#_Toc117859560)

[Insights from exploring Data as Ink 14](#_Toc117859561)

[Making a Drawing Thing 15](#_Toc117859562)

[Technical challenges and approaches 15](#_Toc117859563)

[Initial Version 17](#_Toc117859564)

[A minimal graphical user interface 19](#_Toc117859565)

[Deployment and co-designed changes 19](#_Toc117859566)

[Adding colour 20](#_Toc117859567)

[Adding control 21](#_Toc117859568)

[A continuous canvas 22](#_Toc117859569)

[Adding intentional data capturing — Tinting Data 23](#_Toc117859570)

[Conclusions 25](#_Toc117859571)

[References 26](#_Toc117859572)

[List of Figures 26](#_Toc117859573)

# Introduction

All code relevant to the prototype is openly available online: <https://github.com/opendott-wats/D2.3-Documentation-of-Prototypes>

The main research question informing the work presented in this documentation asks:

* How can creative technologist practice enable nourishing relationships between internet connected technologies and our bodies and sense of self?

The objective of WP2 was to develop capacity to shape emerging best practices around the use of open technology and practices in relation to wearable technologies. To address this objective, I used research through design and a thinking through making[2] approaches as a creative technologist using electronics and code to investigate both material and concepts. I focused on step tracking data as an iconic and well understood data characteristic by investigating, through making, physical digital prototypes which used code and electronic hardware as materials of inquiry. Furthermore, the thinking through making approach also examined my own agency as a researcher towards the material—of sensors, microprocessor prototyping boards, and other digital components related to internet connected technology for the body.

The development of prototypes took place in three major stages:

* [Making a step tracker using open technologies](https://docs.google.com/document/d/1IctPUXXNq_TWvBiSAWoRxdm3bTdO59Oh/edit#bookmark=id.tyjcwt)
* [Exploring ‘Data as Ink’ using creative coding](https://docs.google.com/document/d/1IctPUXXNq_TWvBiSAWoRxdm3bTdO59Oh/edit#bookmark=id.17dp8vu)
* [Making ‘The Drawing Thing’](https://docs.google.com/document/d/1IctPUXXNq_TWvBiSAWoRxdm3bTdO59Oh/edit#bookmark=id.26in1rg)

Based on [insights I gained from the open technology training in WP2](#learning-from-open-technology-training) I decided to focus on step count data as an iconic, well known data characteristic and making step trackers as single function, relatable digital wearables.

To understand the concept of a step tracker I set out to think it through by making one. In parallel, I explored the idea of using ‘data as ink’ by making a simple drawing application and using pre-recorded data. In connection with the co-designed IoT concept ‘Haptic Memories’ from WP1, insights from both lines of inquiry led to the concept: ‘The Drawing Thing’. This final prototype was then deployed with a participant (remotely) and extended collaboratively with them in a co-design activity over the course of 3 months with regular meetings every two to three weeks.

The purpose of making the prototypes, was to shape a design probe which could be given to participants to explore their relationship with data from their bodies. A key result of my explorations was that technology aesthetics and concerns remained in the foreground for the user when using open technology. This makes shaping and developing emerging best practices in relation to internet connected technology which intersects our bodies and sense of self for users and makers challenging.

To capture the way I addressed these challenges, I introduced the term ‘radical minimisation’ to describe my process to assess design considerations during the development of the prototypes. 'Radical minimisation' uses considerate reduction and removal (of e.g., energy use, code, data, attention, or complexity) in an exhaustive, although playful, way to assess how relevant design considerations are to their intensions and goal(s). It aims to find out what is sufficient without constraining the results to an objective pragmatic scope, but to enable a diverse range of possibilities through an in-depth review. Through this process in-depth material inquiries can be created which unpick[4] the complexity of technology at hand. It connects the making to other contexts such as ‘data minimisation’ (Art. 5 and partially 6 GDPR) or the notion of ‘Lean Data Practices’ as suggested by Mozilla[5]. But also emerging fields in information and communication technologies such as 'Low Carbon and Sustainable Computing'[6], 'Permacomputing'[1], or 'Computing within Limits'[3]. Within the development of prototypes, it enabled me to foster a lean technology approach in prototyping for the body.

Additionally, accuracy of sensors and/or algorithms turned out to be a crucial aspect to have a reliable common ground. This influenced the choice of hardware for the final prototype: using a smartphone. While it opposed the initial objective of using open technology, it addressed the issue of accuracy, provided a known and potentially trusted platform for working with participants, and made me review the lived experience of open technology.

## Why focus on step data

The prototypes focused on step tracking or step data mainly for two reasons 1) it is an iconic and commonly known data characteristic of the body; 2) the act of doing steps is tangible and easy to understand. The hypothesis is that by creating new ways of tangible interaction with the data we gain insights on other forms of relating and having agency to data of the body. However, the simplicity of the data turned out not to be easy to implement algorithmically, as I experienced in making a step tracker.

## Learning from Open Technology training

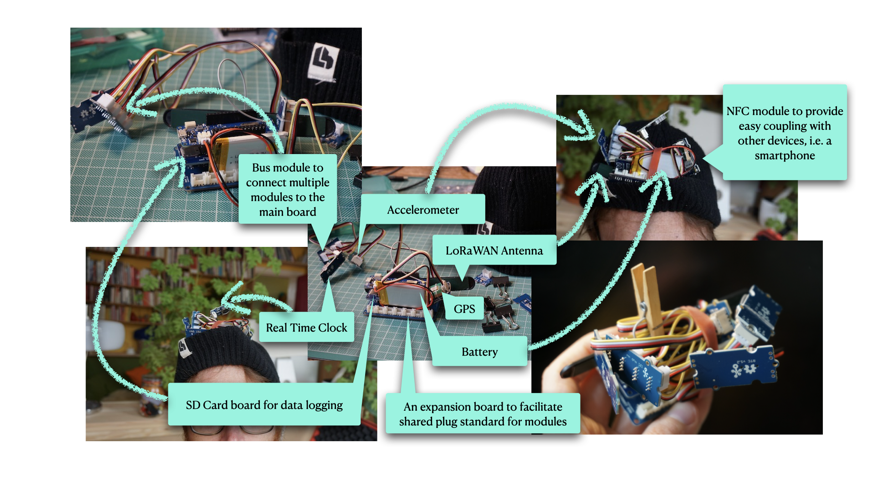


Figure 1: Explorations using Open Technology components during the Training

The key learning from the Open Technology training is that an inquiry into the relationship to data benefits from focusing on minimization of features. During the training I came to the conclusion that a feature packed prototype using novel connectivity leads to technology centred design probes which foreground technological feasibility rather than promoting poetic interactions and in-depth focus on relationships with and through data. Increased risk of technical issues through added complexity would further distract the design process and the work with participants. The aim of the research is to enable a nourishing ground to meaningfully investigate relationships to data from the body, not to demonstrate emerging technologies and their capabilities. Increasingly distributed body data requires an investigation of our relationships, agency, and understandability.

# Making a Step Tracker

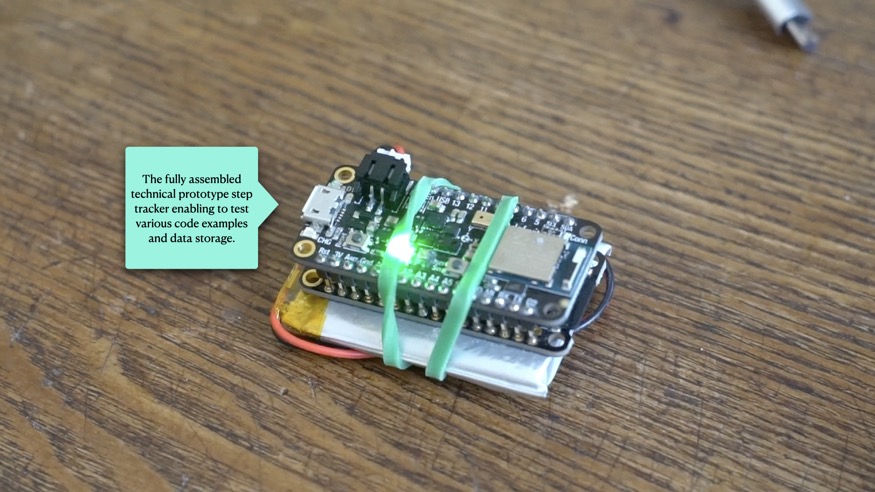


Figure 2: Final Step Tracker used for test runs

Based on the insight related to minimisation and focus on single function in terms of technology, I began to make a step tracker to get to know this kind of wearable with and through material exploration with the following research question:

* Can I make a wearable step tracker using open technologies which is accurate and trustworthy enough to be deployable with participants?

The key aspect I applied to the making was to put myself in the position of someone wanting to make a step tracker at home. My rationale was that it should allow me to assess the understandability of resources that are openly available in contrast to sophisticated industrial products. As a side effect, this also addressed the challenges of making physical digital prototypes at home with constraint facilities during a pandemic situation.

The aim was to think through required components, explore their materiality in hardware and software, and assess how feasible it was in the context of the research to make without becoming an engineering task. For all the sketches I used open technology as much as possible. Firstly, what I had available to begin making as quickly as possible. However, very early in the process of making it became clear that the number of separate components required was too high, resulting in too many wired connections which are prone to break quickly. To address this I set out to find components which integrated features I wanted. Particularly, finding a processing board with a similar set of sensors as the Arduino BLE Sense, that I was currently using, but with an onboard charging unit that would simplify the handling of the power supply. I found the Adafruit Feather Sense board to meet these requirements and offer a better selection of onboard sensors, as explained in section Sensors with built-in step detection.

## Using an online tutorial



Figure 3: Screenshot of the online tutorial I followed initially.

For the first sketch I followed [the most recently published online tutorial on making a step tracker using an Arduino and an accelerometer](https://circuitdigest.com/microcontroller-projects/diy-arduino-pedometer-counting-steps-using-arduino-and-accelerometer) using the Arduino BLE sense board that I had. While the tutorial explained the process well, the way the steps were detected turned out to require a very particular and conscious way of walking, running, or moving - similar to what was shown in the demonstration video of the tutorial.

From this I understood that the detection of steps is more intricate than common sense would assume. There is a technology related gap revealing itself between an act of movement that is so embodied to ourselves, and the challenge of ‘seeing it’ through algorithmic processing of a series of measurements. I was not satisfied with how this algorithm ‘saw’ me walking.

From here I researched new possibilities to improve the accuracy in two directions 1) [Sensors with built-in step detection](#sensors-with-built-in-step-detection), providing the computation/detection of steps; this is known as ‘in silicon’ meaning implementing the computation as part of the integrated circuit which is etched in silicon and thus out of reach to look at for a user or myself as creative technologist. 2) [finding third party code](#finding-third-party-code), preferably open source and preferably within the Arduino community; given the code is open source, the question of understandability here depends on how legible the code is for the users or for myself as creative technologist.

## Sensors with built-in step detection

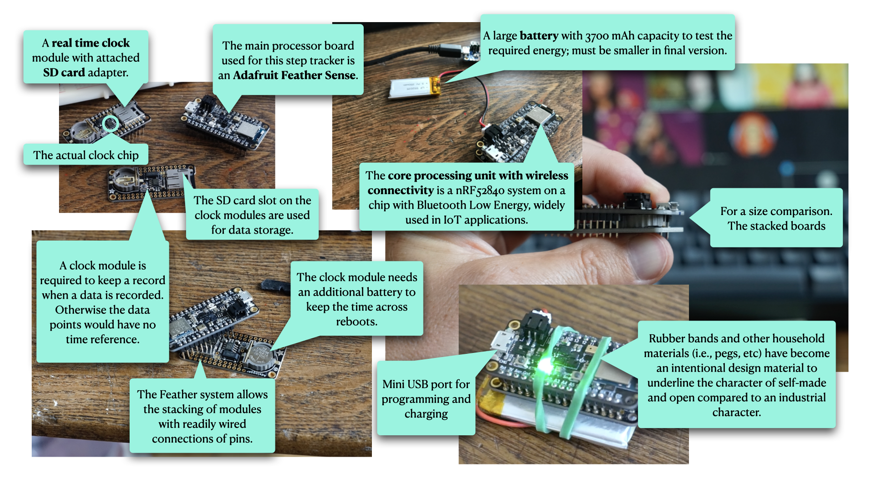


Figure 4: Adafruit Feather Sense based step tracker

The first direction revealed that certain accelerometers provide built-in step detection. However, the one featured on the main processing board I was using did not. I exchanged the Arduino BLE Sense for an Adafruit Feather Sense which featured such a sensor and came with an easy to use code library.



Figure 5: Initial test code for the pedometer feature of the Feather Sense.

This combination of hardware with matching code libraries felt like making progress. I was able to extend the initial testing code with data logging to an SD card. I chose an SD card for ease of use and because is it widely known as an object that stores digital data.

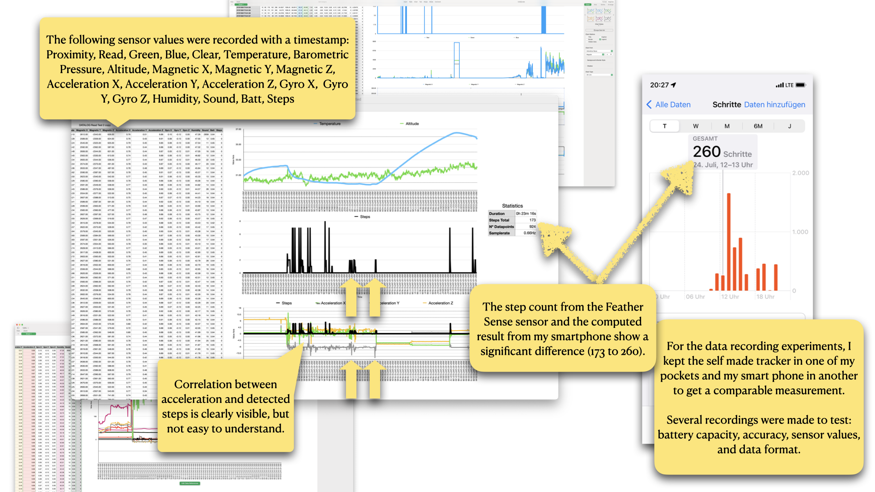


Figure 6: Graphs and initial analysis on recording of the self-made step tracker.

When I took the prototype out for measurements, I also took my smartphone which has built-in step tracking as a reference. After analysing the recordings, it became clear, that the sensor with built-in step detection was not accurate. A gap of roughly 90 steps within 23 minutes.

## Finding third party code

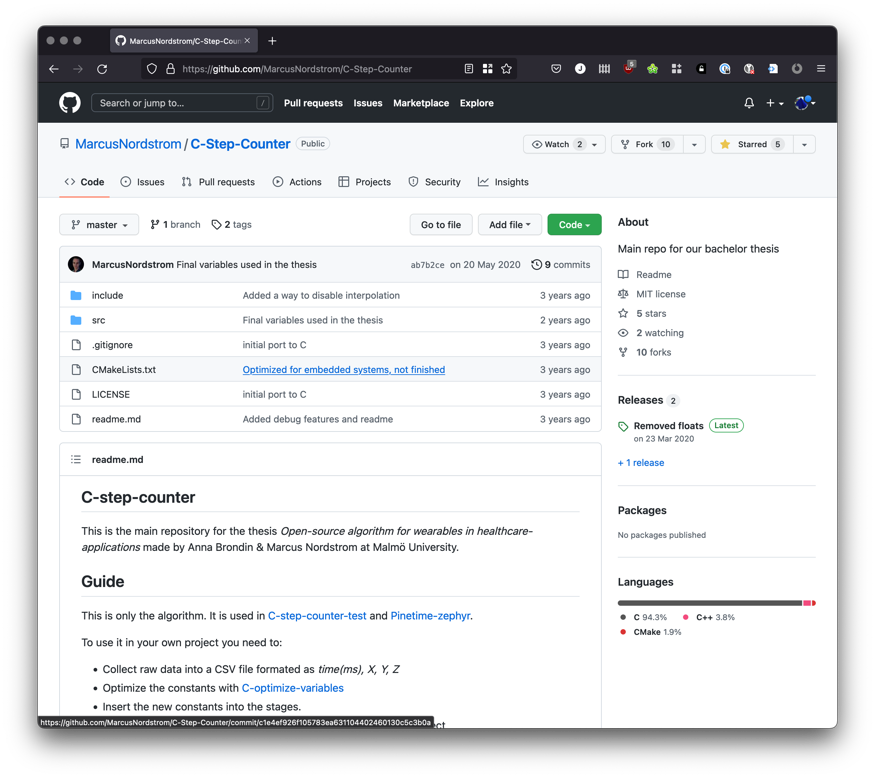


Figure 7: Screenshot of the Open Source C-Step-Counter GitHub repository.

The next attempt I took was to integrate the code library [‘Open Source C-Step-Counter’ by Anna Brondin & Marcus Nordstrom at Malmö University](https://github.com/MarcusNordstrom/C-Step-Counter). The library did not provide a readily useable code sketch for the type of boards I was using. After I had integrated it for the Arduino environment I was able to assess the functionality. The resulting count was very difficult to determine. It was not accurate and rarely worked. I was unable to produce a fully working integration. Other third-party library code was either not available or replicated the simple inaccurate implementation used in the initial online tutorial. Unfortunately, such an outcome is not rare in the context of open source libraries. However, an in-depth discussion of responsibilities of maintainers and authors of code libraries and the reliability of software is out of scope for this documentation.

## Insights from Making a Step Tracker

After also failing with the open source pedometer algorithm, I felt to have reached a pivot point realizing the limitations of open technology on the body. Using open technology related to data of the body has revealed several key challenges: power supply over a longer period of time; directly related to that are code practices, algorithmic choices, and assessment of third-party code for impact on energy use; size of components; and lastly their wearability. The issues of power supply, data storage, and connectivity to other IoT devices are not trivial. And tedious technical fixes felt undermining to trustworthiness of the prototype, especially when deployed to a participant.

In summary, I experienced making a step tracker at home with limited facilities to be relatively easy with prior technical experience with creative technologies. However, the most useable result had its core functionality embedded in a closed piece of technology and was still not accurate compared to an industrial reference product. Hardware issues aside, measurements from test runs showed a significant gap in accuracy when compared to sophisticated step tracking using a smartphone. For the context of the research project on trust and IoT, I consider accuracy a crucial aspect of a prototype. Furthermore, lacking accuracy could undermine building a relationship with the design probe for the participant.

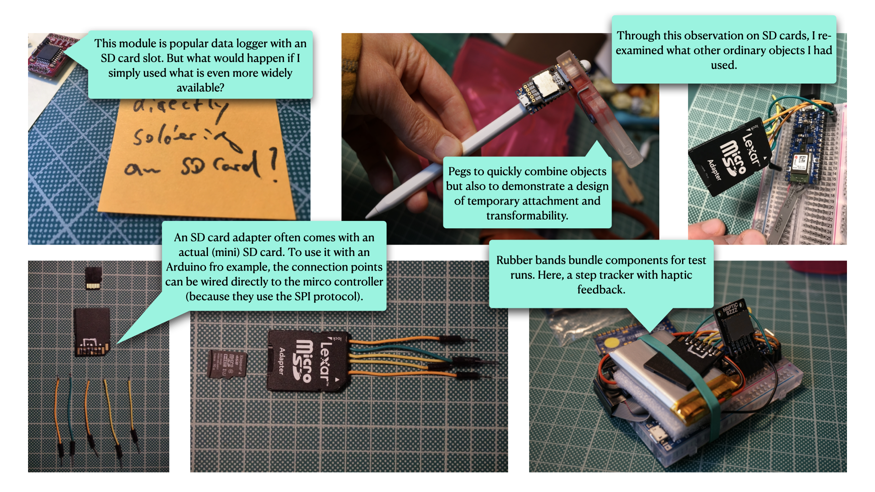


Figure 8: Pegs, rubber bands, and an SD card begin to seep in during the making process.

Pegs, rubber bands, and an SD card began to seep in my use of materials —first unintentionally, later on purpose by soldering wires to an SD card adapter. They began to complement prototype sketches during the making process as ordinary objects of both the known environment and the unknown object. They contributed a handle or an access point that could be recognized, in contrast to an anonymous printed circuit boards (PCB) equipped with oblique tiny components. However, coherent to the guiding minimisation approach they must have a purpose and be functional. The stock SD card adapter with soldered jumper wires for example is no different in its core functionality compared to a component made of bare electronic parts.

# Exploring Data as Ink

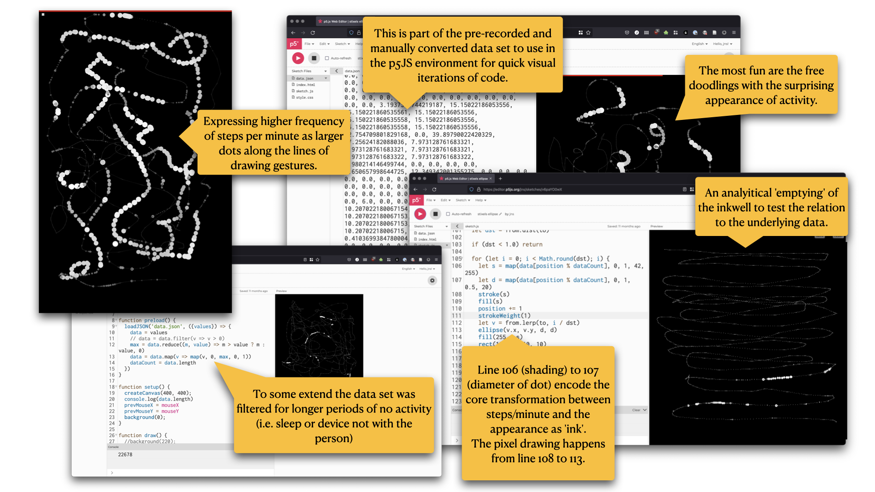


Figure 9: Snapshots from different sketch states using the p5js JavaScript toolkit.

In parallel to making a step tracker, I addressed a research question in connection to the idea of re-tracing data based the concept of 'Haptic Memories' from WP1:

* How can data become a resource for creative expression and reflection?

Picking up the idea of drawing with haptic feedback induced by recordings of step data, I turned the recorded steps into virtual ink for drawing. The data became a resource for creative expression to reflect on the data of one’s body through an embodied act. To prototype this aspect with a visual focus first, I used the creative coding toolkit [p5js](https://p5js.org), a successor to [Processing](http://processing.org), because I prefer using JavaScript for quick iterations and to get a feel for a programming project.

To get a data set of actual step counts, I exported data which my smartphone captured for some time already using an application available on the app store. I picked a time window of 1 day and converted it into JSON format to load into the p5js editor.

The sketch is openly available online: [editor.p5js.org/jns/sketches/v6paYO0wX](https://editor.p5js.org/jns/sketches/v6paYO0wX). Clicking the play button on the top left enables the drawing canvas on the right.

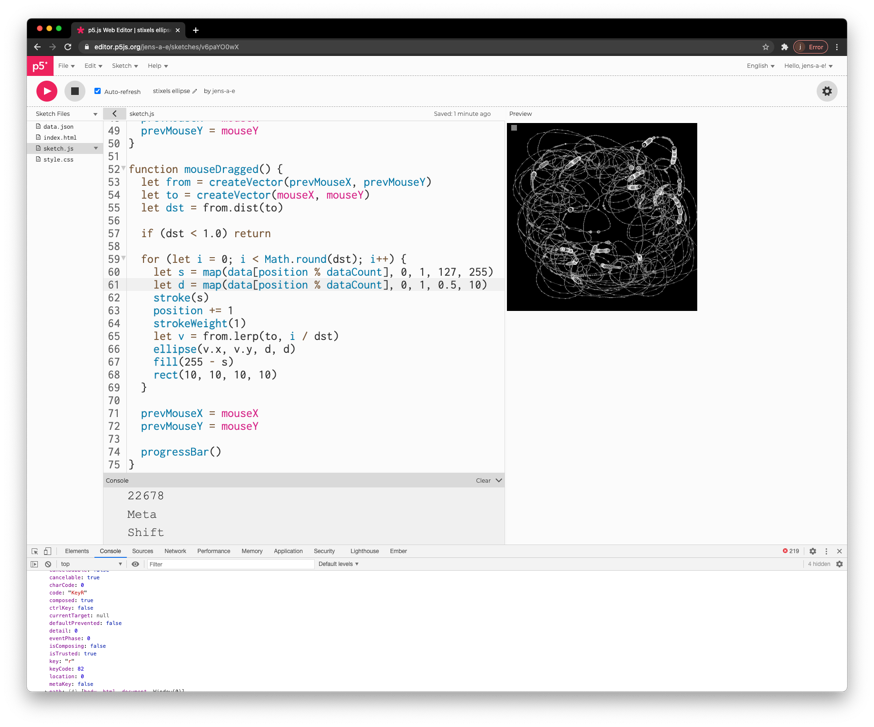


Figure 10: Screenshot of p5js editor with data ink sketch.

## Insights from exploring Data as Ink

In summary, the sketches showed that interaction of data as ink can be a joyful and novel experience is substantial enough to follow up with a deployable prototype.

The act of drawing with data seemed to bring the data back and close to the body as if putting the pen to draw diagrams of data back into the hands of the person from whom the data came from.

The exploration produced three key insights:

1. The interaction creates an unfamiliar but intriguing and meaningful space to reflect on body related data of ourselves.
2. Activated through movement, data as ‘ink’ becomes a visual and tactile collaborative resource, encouraging embodied reflection with and through data *in time*.
3. The purpose of the drawings is to serve as a visual memory aide, a reverberation of a resonance with data of the body. They do not provide an exact representation of the data for analysis.

Key to the prototyping process was to bring together the imagined and the lived experience by putting a conceptual idea of ‘data as ink’ into an interactive sketch. Connecting an actual data set, preliminary data processing, and programming showed that even by simple doodling there was an unexpected joy of anticipating behaviour of the ‘ink’. This opened new ways to reflect and literally re-view data in an embodied way. Does the joyful experience of data as ink also resonate with others than the creative technologist who made it?

# Making a Drawing Thing

The code for the prototype is openly available here: <https://github.com/opendott-wats/D2.3-Documentation-of-Prototypes/tree/main/code/the-drawing-thing>

Making ‘The Drawing Thing’ was the final example of prototypes addressing the question:

* Can we bring the drawing experience closer to the body?

By using a smartphone as a familiar (mundane) hardware platform I addressed the challenges of accuracy and energy management identified earlier (see section Insights from Making a Step Tracker). It is an established hardware platform, is ready at hand in the current situation of working from home, and it is a device familiar to participants. Furthermore, the touchscreen and built-in tactile feedback hardware enable me bring in additional physical closeness to the ‘data as ink’.

For a deployable prototype, the p5js sketches of the previous step had to be ported into a standalone application, native to the operating system (iOS).

## Technical challenges and approaches

Most smartphones feature built-in step tracking. To minimize privacy issues, I used an old Apple iPhone because of the built-in user privacy protection for health-related data:

“The user’s device stores all HealthKit data locally. For security, the device encrypts the HealthKit store when the user locks the device.”[7]

Apple’s on-device database called HealthKit further provided a detailed code library/programming framework which provided to access to the data storage based on the user’s consent:

"Because health data can be sensitive, HealthKit provides users with fine-grained control over the information that apps can share. The user must explicitly grant each app permission to read and[/or] write data to the HealthKit store. Users can grant or deny permission separately for each type of data.“[7]

The prototype benefitted from this feature in two ways: 1) the UI for the consent is controlled by HealthKit, is likely familiar to users/participants and thus likely deemed trustworthy, and 2) that users have to actively consent to individual data characteristics which cannot be side-lined by the application/programmer. Additionally, consent on specific data types which was previously given i.e., for step count data, can be withdrawn by the user independent of the application and without notifying the application:

“To prevent possible information leaks, an app isn’t aware when the user denies permission to read data. From the app’s point of view, no data of that type exists.”[7]

To port the sketches in JavaScript into an application for iOS with access to the HealthKit data store, I learned enough of the platform specific programming language (Swift) and the relevant programming concepts (e.g., provided libraries and frameworks relevant for UI, interaction, image and health data processing). Then I ported the p5js sketches from the previous step into a standalone application, native to the operating system (iOS) which I was able to install and automatically update on the deployed device for the participant.

For initial sketches I used my personal phone and for the deployment a separate re-used device which I made as fully self-contained as possible by doing a: factory reset, removing any pre-installed applications, features, and settings as much as possible, and by creating a new separate Apple account for each device.

Furthermore, the smartphone as a hardware platform provided sophisticated power management, it would record the steps efficiently even when a phone is not active, and lastly featured a touch screen for the drawing canvas. These aspects made it a good candidate to use as a prototype base that focused on the experience of drawing with data in a reliable environment. While, the smartphone is not ‘wearable’ as such, I preferred to keep the aspect the prototype being a single, literal ‘a drawing thing’—highly self-contained tracking and drawing surface. Additionally, connecting a wearable (self-made or ready-made) is not a trivial technical implementation task which raises the potential for errors in the prototype. Specifically in the situation of deploying the prototype remotely, I was conscious to keep this potential as low as possible as pragmatic precaution based on experience. The smartphone based prototype provided the focus on the relationship with and through data, despite sacrificing aspects of wearability. It remains a companion object that is carried about in the body in different personal ways. However, such a wearable, e.g., a fitness armband, would not have provided an integrated drawing surface. Nevertheless, prototypes which are more ‘wearable’ can be explored in future work based on the insights on relating with and through data of the body through drawing. For example, a simple next step is to add a ready-made step tracker which integrates with the health database on the smartphone.

## Initial Version



Figure 11: Testing the initial version.

The design goal of the initial version was to use and show as little technology and capability as possible. Because the device was black, the canvas was black which resulted in a consistent solid appearance of the device when the canvas was empty; the infamous black brick.

The structure of the programming interface of the underlying data storage (Apple HealthKit) to retrieve data, two time-related parameters had to be provided: 1) a start and end date, marking the time window for which to request data, and 2) an interval in which the measurements will be summarized. After trying different intervals, steps per minute revealed the best balance between granularity and amount of resulting ink.

The core data to ink drawing algorithm can be simplified as:

1. On start-up and reset, fill the ‘inkwell’ by querying the data store for step count with an interval of 1 minute since X about days (defined by the user in the settings).
2. When drawing, pick the next steps per minute measurement from the list and use it to adjust stroke thickness and transparency.
3. Stop the drawing functionality when the inkwell is empty.

Based on the former sketches the first data to ink algorithm only produced black and white strokes based on the fact if the current measurement was greater than zero or not; a direct digital interpretation. In a second step the variance of the data (the fluctuating steps per minute) became the basis to influence stroke opacity/transparency.

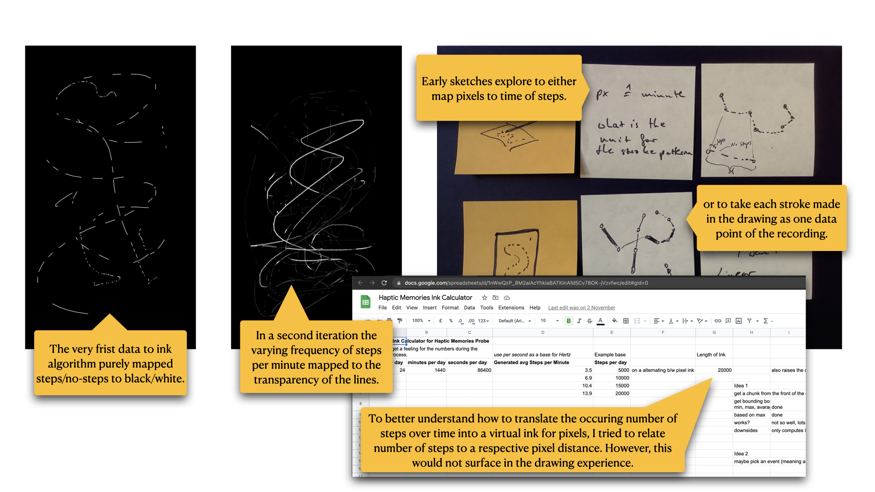


Figure 12: Screenshots of drawings and sketches for the data to ink conversion.

## A minimal graphical user interface

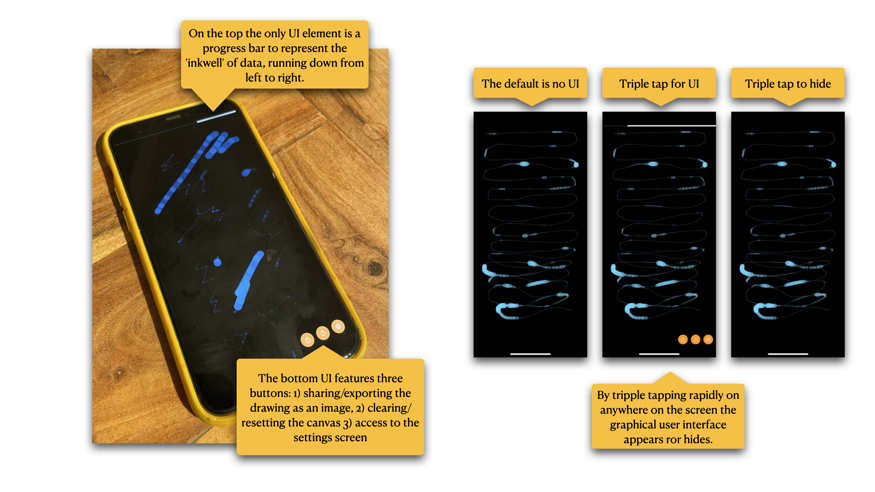


Figure 13: The interface active, the process of activation shown in screenshots.

The very first iterations of the prototype had no graphical user interface. However, to enable sharing of drawings in conventional ways, a manual clearing of the canvas, and access to the settings for stroke behaviour and how many days of data to consider, I added a minimal UI.

* A progress bar to show the state of the inkwell
* A set of three buttons for sharing/exporting the drawing, clearing the canvas, and accessing the settings.

At a later stage in the co-design process with the participant, The UI was hidden by default to give the drawing space and activity undistracted attention from the start. By triple tapping on the screen the UI appears and can be hidden with the same gesture. The triple tap made it easy to integrate in code but more importantly, followed through with the ‘radical minimisation’ approach by shifting interactions to gestures and movement and to avoid adding a button UI despite the intention of removing/hiding UI. This functionality needed to be explained to the participant(s) to anticipate potential confusion that is not intentional.

## Deployment and co-designed changes

In the initial minimal version, the prototype was deployed with a participant and extended collaboratively with them in a co-design activity with regular meetings every two to three weeks. The participant is a creative practitioner in the fields of music and illustration. Over the course of the 3 months during which the participant lived with the device they produced 1380 drawings in total.

The following section documents the iterative changes made based on the conversations during the co-design activity. In summary four adaptation have been made to the prototype:

* [Adding colour](#_Adding_colour) to the stroke based on time of the current data point.
* A [settings screen](#adding-control) to give the participant more control on the spectrum in which the stroke width varies and the number of days from which to transform steps to ink.
* [A continuous canvas](#a-continuous-canvas) to create drawings which can stretch over multiple drawing sessions; fading strokes enable potentially infinite drawings.
* Adding the ability of [intentional tinting of data](#X4a80d2eee945d8cd5386d42b14ba4ef1bb3d075) by capturing colour hue from the environment like taking a photograph.

### Adding colour

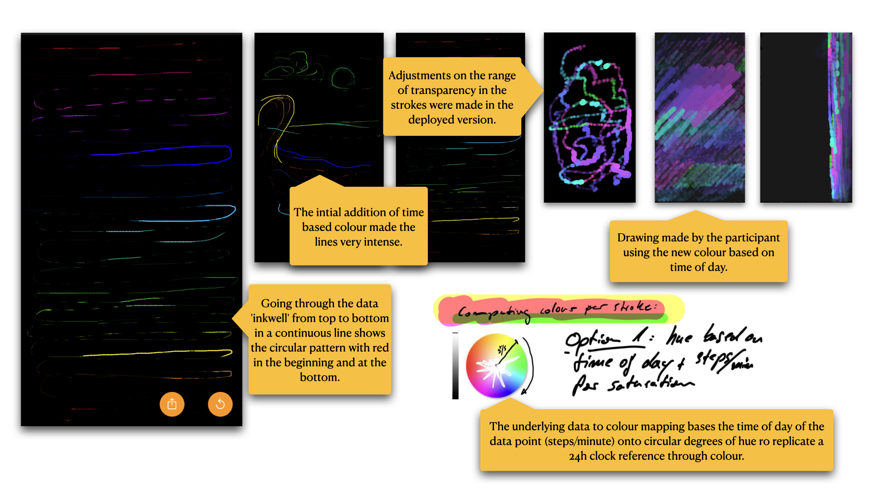


Figure 14: Initial introduction of colour to the stroke based on time of day of the data point.

After about 3 weeks with the initial version, the first request of the participant was to have colour in the ink. I responded to this by transforming the timestamp of the current steps/minute data point to a hue value. By normalizing the time of day on a 24 hours basis to values between 0.0 to 1.0 a colour could be computed where the hue represents the minute of day the steps took place. This worked well for some time but was described by the participant as repetitive after a while. However, having colour was appreciated and followed up with a later change of [manual intentional colour recording](#X4a80d2eee945d8cd5386d42b14ba4ef1bb3d075).

### Adding control

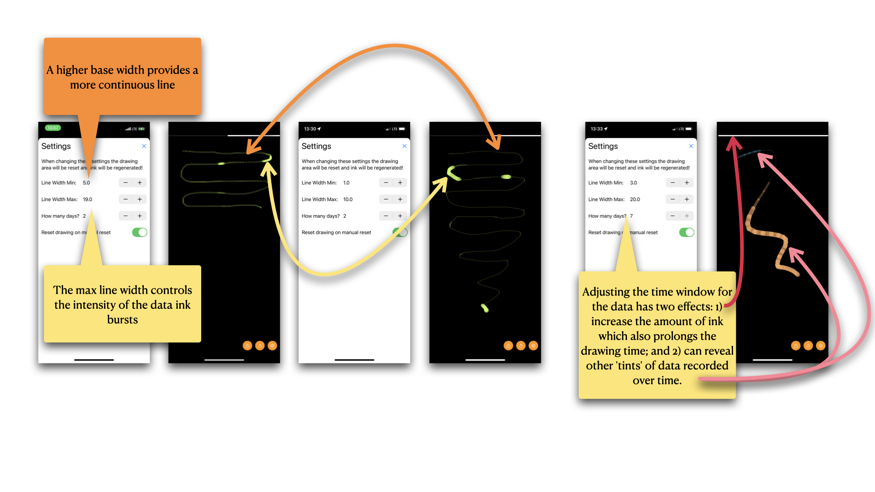


Figure 15: Showing the connections between the settings screen and the change in drawing.

To give the participant control of how the stroke width could vary and to adjust the amount of ink, I introduced a settings screen providing access to those parameters. Over time the participant actively incorporated these controls in their drawing practice. Based on my experience the implementation of such user interface controls can outweigh the code of the core functionality of the application extensively. Fortunately, linking the parameters of the settings to their respective locations in the drawing code did not add too much additional code because of a direct support by the programming framework.

### A continuous canvas

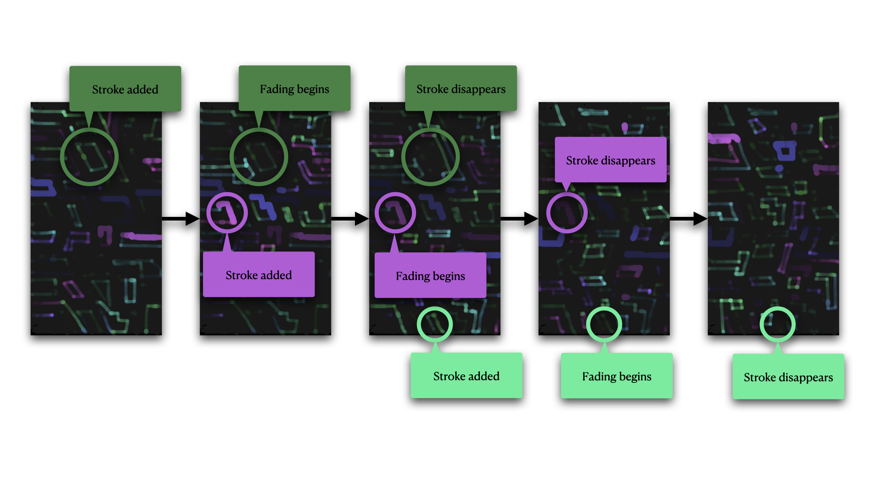


Figure 16: The effect of fading strokes over time on the continuous canvas.

To open new ways to expand the drawing with data as ink the idea emerged to change the canvas for longer periods of drawing. By continuously fading strokes over time the canvas became an infinite space clearing itself over time. Interestingly it also added depth to the drawings as a line would appear further away the ‘older’ it became. This effect was not anticipated. The manual reset of the canvas was not removed to still be able to clear the canvas. However, the conversation with the participant touched on introducing other constraints such as having to wait for hours or days until the canvas would clear by itself. Or, to play with enforcing a timely distance between the reflections and give ‘The Drawing Thing’ a character of its own.

### Adding intentional data capturing — Tinting Data

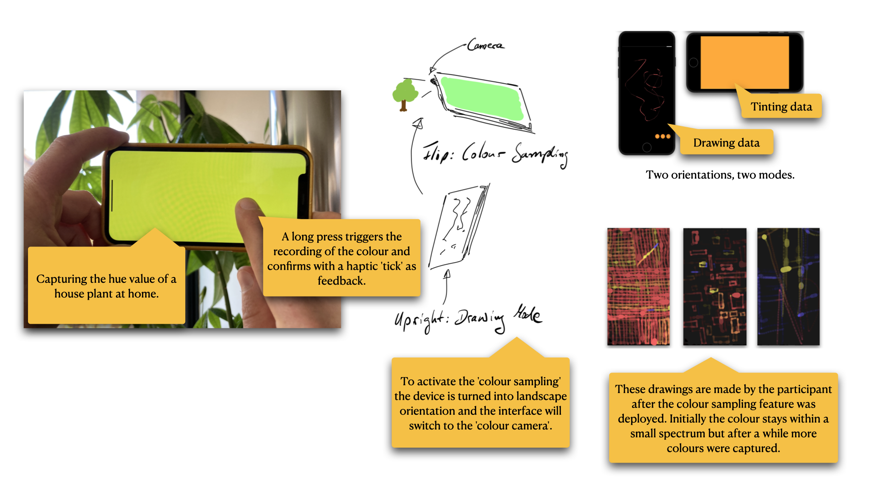


Figure 17: How to activate the ‘colour camera’ and drawings by the participant.

Based on the [initial introduction of colour](#adding-colour) the last change that was made in collaboration with the participant was to move the determination of the colour value away from a deduction based on the data. By introducing an intentional way to influence—or ‘tint’— the data we were interested whether the manual recording of data would provide an additional quality to the relationship to the step data; and if this would emerge meaningfully in the drawings. The ‘colour camera mode’ activated by tilting the device into landscape, moving it around to search for a hue value worth recording, and through a long press on the screen ‘locking’ this colour resulting in a tinting of any data that is recorded until the next colour is recorded.



Figure 18: Gradually reducing the camera feed; from blurs to a single pixel (from left to right).

The implementation of a manual recording of colour became challenging because a programming interface narrowly designed for the purpose of taking a picture obstructs simple access to the image stream of a built-in camera without showing the picture or manipulating it. To allow for a real-time hue extraction, nevertheless, I had to extensively research and add complex code which allowed me to analyse the incoming pixel stream of the camera in real-time. After removing this technical obstacle, I was able to explore making a ‘colour camera’ (see Figure 18 above).

Over several design stages, it became clear that key to the design process was to playfully follow the minimization approach and follow the most radical design considerations. Before I summarise the steps, it is important to note that all changes made to the prototype have been done in code only. This was necessary to deploy the changes as an update remotely without requiring the participant to physically repeat the changes to their drawing thing object. As an aside mixed forms of physical and software changes potentially elicit further insight and present opportunities be explored in future work.

The initial idea was simple: use the built-in camera to let the user take a picture, extract the most dominant colour (find or write an algorithm to do this). However, related to ‘privacy by design’ I wanted to avoid storing pictures on the device—Instead, the final data stored on the device was reduced to a timestamp (for correlating with the step data) and the hue value (not even the full colour value with lightness and saturation).

The next step was to minimise the resemblance of a photo camera, I experimented with defamiliarising the life view of the camera. As already pointed out, this posed technical hurdles which prolonged the design process and deepened the reflection on the camera vision as a source for a colour. Then, I blurred the image camera by setting the focus of the camera to as close to the lens itself as possible and the opening the aperture (see Figure 18 left). Next I removed the blur because conceptually the radiance of the blur would produce additional colours which are not present in the source image. I removed it by reducing the resolution of the camera feed. This was only possible to a degree while still allowing a user to identify what is shown in the image. Adding a filter to the image stream, in order to pixelate the image in varying degrees, finally allowed me to reduce the resolution to a level which began to make the camera feel like an abstract sensor (see Figure 18 middle).

I explored my surroundings with this new extreme low-resolution camera for a while. By getting a feel for this new ‘sensor’ and to follow through on the ‘playful radical minimalisation’, I took the leap to reduce the camera view to a single pixel resolution with only its hue value because it is the only data which is actually relevant (see Figure 18 right). Left with either showing the hue value or nothing, the end result is a solid-coloured screen which is operated by tilting the device and a long press to ‘lock’ the colour. This ‘single pixel camera’ resonated not only with the minimization approach but also addresses ‘data minimisation’ (Art. 5 and partially 6 GDPR) and privacy as ‘data protection through design’ (Art. 25 GDPR) by asking what is essential for the data recording.

# Conclusions

This documentation has detailed and demonstrated the challenges when making functioning interactive prototypes in the context of wearables through a range of different prototypes. I navigated the technological difficulties by understanding them as a material quality. Instead of being distracted by features and technological capabilities, my approach began to ask what can be removed. For example, by technological functions, sensors, and novel connectivity from the list of aspects to integrate. By instead focusing on a single data type (step count) as a representative characteristic of data of the body, I was able to create an in-depth inquiry into what the relationship to data of the body reveals when experienced as a drawing activity/tool/experience.

# References

1. Ville-Matias Heikkilä. 2020. Permacomputing. Retrieved June 7, 2021 from http://viznut.fi/texts-en/permacomputing.html.

2. Tim Ingold. 2013. Making: Anthropology, Archaeology, Art and Architecture. Routledge, London.

3. Marloes de Valk. 2021. A pluriverse of local worlds: a review of Computing within Limits related terminology and practices. *In LIMITS ’21: Workshop on Computing within Limits*.

4. Jayne Wallace and Patrick Olivier. 2010. Unpicking The Digital. *Momentum pp 8 - 11*.

5. Lean Data Practices. *Mozilla*. Retrieved October 24, 2022 from https://www.mozilla.org/en-US/about/policy/lean-data/.

6. Low-carbon and sustainable computing. *Wim Vanderbauwhede*. Retrieved October 25, 2022 from http://dcs.gla.ac.uk/~wim//low-carbon-computing/#.

7. Protecting User Privacy | Apple Developer Documentation. Retrieved October 24, 2022 from https://developer.apple.com/documentation/healthkit/protecting\_user\_privacy.

# List of Figures

[Figure 1: Explorations using Open Technology components during the Training 5](#_Toc117858904)

[Figure 2: Final Step Tracker used for test runs 6](#_Toc117858905)

[Figure 3: Screenshot of the online tutorial I followed initially. 7](#_Toc117858906)

[Figure 4: Adafruit Feather Sense based step tracker 8](#_Toc117858907)

[Figure 5: Initial test code for the pedometer feature of the Feather Sense. 9](#_Toc117858908)

[Figure 6: Graphs and initial analysis on recording of the self-made step tracker. 10](#_Toc117858909)

[Figure 7: Screenshot of the Open Source C-Step-Counter GitHub repository. 11](#_Toc117858910)

[Figure 8: Pegs, rubber bands, and an SD card begin to seep in during the making process. 12](#_Toc117858911)

[Figure 9: Snapshots from different sketch states using the p5js JavaScript toolkit. 13](#_Toc117858912)

[Figure 10: Screenshot of p5js editor with data ink sketch. 14](#_Toc117858913)

[Figure 11: Testing the initial version. 17](#_Toc117858914)

[Figure 12: Screenshots of drawings and sketches for the data to ink conversion. 18](#_Toc117858915)

[Figure 13: The interface active, the process of activation shown in screenshots. 19](#_Toc117858916)

[Figure 14: Initial introduction of colour to the stroke based on time of day of the data point. 20](#_Toc117858917)

[Figure 15: Showing the connections between the settings screen and the change in drawing. 21](#_Toc117858918)

[Figure 16: The effect of fading strokes over time on the continuous canvas. 22](#_Toc117858919)

[Figure 17: How to activate the ‘colour camera’ and drawings by the participant. 23](#_Toc117858920)

[Figure 18: Gradually reducing the camera feed; from blurs to a single pixel (from left to right). 24](#_Toc117858921)