

# Diaries

Paolo Bettelini

## Contents

<b>1</b>	<b>Diaries</b>	<b>2</b>
1.1	2022-12-13 . . . . .	2
1.2	2022-12-14 . . . . .	2
1.3	2022-12-15 . . . . .	3
1.4	2022-12-20 . . . . .	3
1.5	2022-12-21 . . . . .	3
1.6	2022-12-22 . . . . .	3
1.7	2023-01-09 . . . . .	4
1.8	2023-01-10 . . . . .	4
1.9	2023-01-11 . . . . .	4
1.10	2023-01-12 . . . . .	5
1.11	2023-01-17 . . . . .	5
1.12	2023-01-18 . . . . .	5
1.13	2023-01-19 . . . . .	6
1.14	2023-01-23 . . . . .	6
1.15	2023-01-24 . . . . .	6
1.16	2023-01-25 . . . . .	7
1.17	2023-01-26 . . . . .	7
1.18	2023-01-27 . . . . .	7
1.19	2023-01-30 . . . . .	8
1.20	2023-01-31 . . . . .	8
1.21	2023-02-01 . . . . .	9
1.22	2023-02-02 . . . . .	9
1.23	2023-02-03 . . . . .	10
1.24	2023-02-06 . . . . .	10
1.25	2023-02-07 . . . . .	10
1.26	2023-02-08 . . . . .	10
1.27	2023-02-09 . . . . .	11
1.28	2023-02-10 . . . . .	11
1.29	2023-02-13 . . . . .	11
1.30	2023-02-14 . . . . .	11
1.31	2023-02-15 . . . . .	11
1.32	2023-02-16 . . . . .	12
1.33	2023-02-17 . . . . .	12

# 1 Diaries

## 1.1 2022-12-13

Work hours:

**08:20 - 11:15:** Setup and research

Today I setup the git repository with the initial files (CLI tool, documentation, etc.). I then spent the rest of the working session researching the topic of the project.

The planning has yet to be done. The goal for the next working session is to make the Gantt chart.

## 1.2 2022-12-14

Work hours:

**15:10 - 16:20:** Alpha matting test

**16:00 - 16:20:** Gantt chart

Today I started using the `opencv` library (The Rust wrapper). I used the sample images (target and trimap) from [https://docs.opencv.org/4.x/dd/d0e/tutorial\\_alphamat.html](https://docs.opencv.org/4.x/dd/d0e/tutorial_alphamat.html). The output is not correct, but it shows some matting to be applied.



However, I noticed a small problem: when the input images are wrong, instead of returning `Result::Err` (as per Rust-philosophy), it dumps the core.

I also started making the Gantt chart.

### 1.3 2022-12-15

Work hours:

**08:20 - 09:50:** CLI executable

**10:20 - 11:20:** Gantt chart

Today I fixed the trimap matting. The trimap was being read using `IMREAD_COLOR` rather than `IMREAD_GRAYSCALE`. I then added the first CLI arguments to the executabl (target image, trimap image, output image). I need to find a solution to avoid the core dumps from `opencv`.



In the second half of the working session I completed the gantt chart

### 1.4 2022-12-20

Work hours:

**08:20 - 09:50:** OpenCV documentation

**10:20 - 11:20:** Documentation and requirements

In the first half of the working session I read some documentation about OpenCV. The goal is to be able to remove the background of an image given its trimap. In the second half I continued setting up the documentation (requirements and layout).

### 1.5 2022-12-21

Work hours:

**15:05 - 15:30:** Documentation

Today I continued the analysis section.

### 1.6 2022-12-22

Work hours:

**08:20 - 09:50:** Background removal research

**10:05 - 10:30:** Read the review of my previous project documentation

**10:30 - 11:20:** GUI Research

Today I did not write any actual code, however, I did some research on which technologies I will need to use. I realized that the background removal cannot be done using `opencv`, but rather with another library. Not much effort is necessary in order to finish the CLI tool, so I started looking into the Rust GUI libraries. I also continued writing the requirements in the analysis section.

The plan for the next working session is to finish the CLI tool.

## 1.7 2023-01-09

Work hours:

**13:15 - 16:30:** CLI

Within the last week I completed the CLI tool. The CLI tool is capable of generating soft masks given trimaps. It can also apply operations to images such as: making the background transparent, filling the background with a color or replacing the background with another image.

### Matting CLI

```
Usage: matting-cli [OPTIONS] --target <TARGET> [--mask <MASK>|--trimap <
      TRIMAP>]
```

#### Options:

<code>-i, --target &lt;TARGET&gt;</code>	Target image
<code>--mask &lt;MASK&gt;</code>	Background mask image
<code>--trimap &lt;TRIMAP&gt;</code>	Trimap image
<code>--save-mask &lt;SAVE_MASK&gt;</code>	Save mask path
<code>-o, --output &lt;OUTPUT&gt;</code>	Output image
<code>-f, --fill &lt;FILL&gt;</code>	Fill background action
<code>-t, --transparent</code>	Transparent background action
<code>-r, --replace &lt;REPLACE&gt;</code>	Replace background action
<code>-h, --help</code>	Print help information
<code>-V, --version</code>	Print version information

The CLI tool still lacks error handling.

Another feature to add is the `--verbose` flag, which prints what the program is doing along with timestamps.

## 1.8 2023-01-10

Work hours:

**08:20 - 9:50:** Error handling

**10:05 - 11:00:** Verbose flag

Today I handled every possible crash in the CLI tool. Whenever an error occurs it prints the according message. I also started implementing the `--verbose` flag, which prints program statuses and timestamps.

The plan for the next working session is to finish implementing the verbose flag. Next, the server application can be developed.

## 1.9 2023-01-11

Work hours:

**15:00 - 16:20:** Verbose flag and refactor

Today I continued implementing the `--verbose` flag. The CLI tool now prints what it is doing, but I haven't implemented the timestamps yet. I also did some refactoring of the code and cleanup.

## 1.10 2023-01-12

Work hours:

**08:30 - 09:50:** CLI timestamps

**10:05 - 11:20:** Documentation

The CLI now fully supports the `--verbose` flag and prints the elapsed time for each operation.

```
> matting-cli -i target.jpg --trimap trimap.png --save-mask mask.png
--verbose -o out.png --fill red
```

```
Reading target image... Done! [4.021485ms]
Reading trimap image... Done! [1.976477ms]
Generating soft mask... Done! [7.104532642s]
Reading target image... Done! [178.884124ms]
Saving soft mask... Done! [509.050896ms]
Filling background with color... Done! [117.90393ms]
Saving output... Done! [954.085622ms]
```

I created the `log!()` macro which executes an expression and logs a message and the elapsed time if a given flag is true.

```
let mask = log!(
  "Generating soft mask",
  args.verbose,
  matting::generate_mask(&target, &trimap)? // heavy lifting
);
```

In the second half of the working session I continued the documentation. I did not write any actual documentation, just boilerplate and useful stuff that I'll be using.

The plan for the next working session is to crate the server worker.

## 1.11 2023-01-17

Work hours:

**08:30 - 11:20:** Server and API

Today I starting implementing the backend worker to process the images. The route still does not work, although the development of the webserver should go smoothly after this issue.

The plan for the next working session is to continue the documentation.

## 1.12 2023-01-18

Work hours:

**15:00 - 16:20:** Documentation

Today I continued the documentation. I wrote the following sections:

1. Introduction/Information
2. Trimap Matting

### 1.13 2023-01-19

Work hours:

**08:30 - 10:40:** Webserver

**10:40 - 11:00:** Helped a classmate

**11:00 - 11:20:** Documentation

Today I continued working on the backend. The webserver now serves the static websites files. The upload root can read the image data.

I also continued the documentation and wrote a small section about OpenCV ([OpenCV](#)).

### 1.14 2023-01-23

Work hours:

**13:20 - 16:20:** Webserver and frontend

Today I continued working on the backend and the frontend. The backend successfully receives POST requests containing an image and can read its data. The frontend can load images onto the canvas and send the canvas contents to the server. I also started implementing the brush feature. The canvas is already capable of simple painting features.

### 1.15 2023-01-24

Work hours:

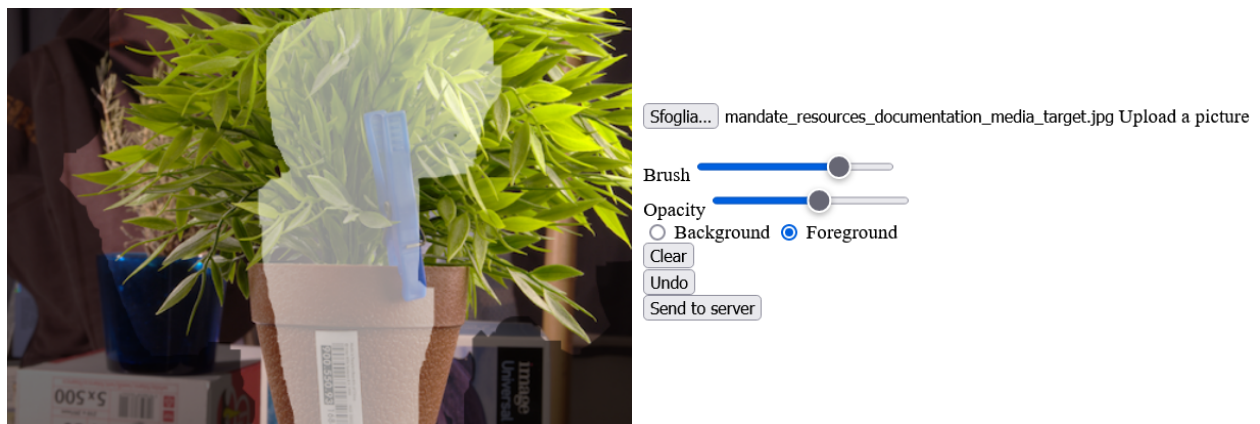
**08:30 - 11:20:** Frontend Trimap Painting

Today I focused on implementing the painting feature on the frontend.

When an image is uploaded, I can start painting over it. There are a few settings and features such as:

- Brush size slider
- Background / Foreground selection
- Opacity slider
- Undo action
- Clear canvas action

I still need to implement the redo feature, and the undo features does not work if the clear feature has been used previously. The plan for the next working session is to solely continue the documentation.



## 1.16 2023-01-25

Work hours:

**09:05 - 10:30:** Documentation

Today I continued the documentation and wrote the following sections

- CLI/Compilation
- CLI/Usage
- CLI/Examples

## 1.17 2023-01-26

Work hours:

**08:30 - 10:50:** Backend

**10:50 - 11:10:** Helped a classmate

**11:10 - 11:25:** Backend

Today I continued the backend code. The website now sends both the **target** and **trimap** image. In the future it could just send **target** and **mask**. The backend is able to read the data image and convert it to the **opencv** types. Everything is ready to implement the processing logic on the backend.

## 1.18 2023-01-27

Work hours:

**08:30 - 09:50:** Backend

**12:30 - 13:50:** Backend and frontend

Today I continued the backend logic and frontend UI. The server is still not able to generate a mask given the trimap. In order for it to work the trimap must be processed (transparent pixels → gray ones).

For now, the trimap is treated as the actual mask. The server removes the background and returns the image to the client, which displays it on the web page.

The goal for the next session is to look into pre-processing the trimap and continuing the web UI.

## 1.19 2023-01-30

Work hours:

**13:15 - 13:30:** Helped a classmate

**13:30 - 16:20:** Frontend

Today I focused on the frontend website. I'm using the **tailwind** CSS framework to style the page. So far the page is still a very early draft.

I plan to structure the website as follows:

1. **Introductory section:** an explanation about how the process works
2. **Upload image and trimap section:** uploading from and trimap drawing tools
3. **Mask section:** a place to see the generated or uploaded alpha mattes mask
4. **Transformation section:** a place to select the operation for the background
5. **Result section:** final result display

## 1.20 2023-01-31

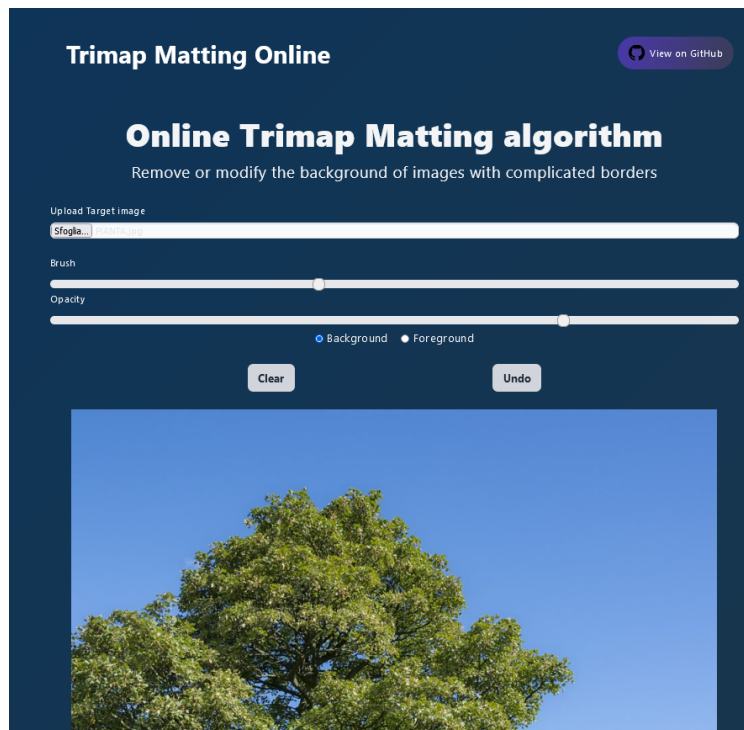
Work hours:

**08:20 - 11:20:** Frontend

Today I continued the website code. The design is much better and I fixed a couple of bugs:

- The painting brush would draw weird spikes when moving slow
- The browser would keep the old input values upon a page refresh

The website now looks as follows:





## 1.21 2023-02-01

Work hours:

**10:50 - 12:0:** Frontend and backend

Today I started implementing the background *transformation* feature on the website. I added a form to choose between **transparency**, **fill with color** or **replace with image**. I created the endpoint for this request and started implementing the logic.

The plan for the next working session is to hopefully make this feature work.

## 1.22 2023-02-02

Work hours:

**08:20 - 11:40:** Frontend and backend

**11:40 - 12:10:** Frontend

Today I implemented the background operation selection on the website. The user can choose wheter to replace the background with an image, fill it with a color or leave it transparent.

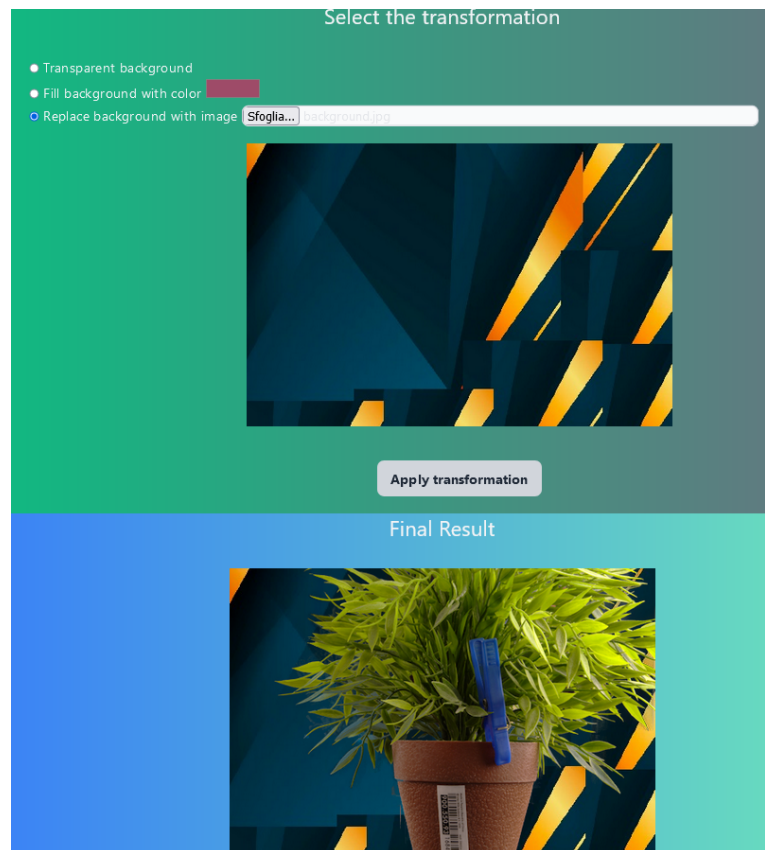
I ended up creating 3 routes for each transformation. The routes now are

- **/api/matting**
- **/api/fill/color**
- **/api/transparent**
- **/api/replace**

Both the backend and frontend code are a bit messy and will definitely need some refactoring.

I also continued the page design a bit.

The plan for the next working session is to continue the documentation.



## 1.23 2023-02-03

Work hours:

**08:20 - 09:50:** Documentation

**14:20 - 15:00:** Trimap processing.

Today I continued the documentation.

- Completed the `CLI/Examples` section
- Added missing citations
- Made the `Implementation` section(s)
- Fixed various typos
- Continued the `Trimap Matting` section

In the remaining time I started looking into preprocessing the client-sent trimap. I need to remove the alpha channel and replace the transparent pixels with a gray color.

## 1.24 2023-02-06

Work hours:

**13:15 - 15:30:** Frontend

**15:40 - 16:30:** Research

Today I finally made the mask generation work. I rewrote the trimap painting system a bit such that the *border* is a color (gray). However, the entire canvas is colored gray by default.

The fundamental remaining things to do are the following:

- Better website design (with a loading animation)
- Better error handling (avoid core dumps and properly respond)

In the remaining time I did some research on how I could use the `rayon` crate to add multithreading to my image processing functions.

## 1.25 2023-02-07

Work hours:

**08:20 - 11:20:** Frontend

Today I continued the frontend. I added some logic in order to enable inputs/buttons only when they can be used. I added the download button under the mask image and the final result. I also refactored some code and commented it.

## 1.26 2023-02-08

Work hours:

**08:20 - 11:20:** Discussed

**11:20 - 12:50:** Implemented changes.

Today I discussed with my supervisor about core dumps of the `opencv` library and some error handling. I started implementing some minor functionality changes in the CLI program.

## 1.27 2023-02-09

Work hours:

**08:20 - 11:00:** CLI tool

Today I made some changes to the CLI tool. Namely, if the `--output` parameter is not specified then it will generate a name with the current timestamp (like `target_20230209-09:52:23.jpg`). If the user wants to save an image with transparency as something other than a `png`, he will be warned about it.

In the second half of the working session I implemented the size checks, refactored some code, did some tests and then read the specific requirements of the project.

## 1.28 2023-02-10

Work hours:

**08:20 - 09:50:** Documentation

**12:30 - 13:40:** Documentation

Today I continued the documentation. I created or continued the following sections:

- Glossary
- `OpenCV.Rust` Binding
- `OpenCV.Matting` library

I also fixed the default filename with the timestamp (es. `target_20230209-095223.jpg`).

## 1.29 2023-02-13

Work hours:

**13:15 - 16:15:** Frontend

Today I improved the frontend design (composition, titles, colors). I fixed the autocomplete behavior of the inputs. I also implemented the same-size check on the backend transformation routes.

## 1.30 2023-02-14

Work hours:

**08:20 - 11:20:** Frontend

Today I kept developing the frontend page.

- I implemented the image-size checks on the frontend.
- I added a loading gif when an image is processing.
- Better design and colors.

The plan for the next working session is to implement the bucket painting tool. I figured it'd be very useful.

## 1.31 2023-02-15

Work hours:

**10:35 - 12:10:** Bucket tool

Today I started implementing the bucket/fill tool. The algorithm works and I just need to implement it as a feature.

## 1.32 2023-02-16

Work hours:

**08:30 - 11:25:** Bucket tool

Today I finished implementing the fill/bucket tool. I fixed some bugs regarding this tool, for instance, sometimes it would go into an infinite loop. I also added a footer to the website.

Although the fill tool works, when an enclosed area is filled, a thin layer of pixels of the color that's being replaced would remain at the border of the enclosed area. This is due to sub-pixel rendering/anti-aliasing. It would be best both for the trimap and the flood fill not to have anti-aliasing. I could not turn it off and instead implemented a more tolerant algorithm which only checks if the nearest pixel is one of the two colors (out of three) which are not being used to fill, rather than checking if the color is not equal to the one of the fillage.

The plan for the next working session is to continue the documentation and try to turn off the antialiasing.

## 1.33 2023-02-17

Work hours:

**08:30 - 09:50:** Documentation

**12:30 - 13:30:** Documentation

**13:30 - 14:00:** Frontend

Today I continued the documentation. I created or modified the following sections

- `OpenCV.Trimap.colors`
- `Implementation.Website.Drawing.mechanics`
- `Implementation.Website.Flood.fill`
- `Implementation.Website.Undo.feature`

I also tried to disable the anti-aliasing but the smoothing is still there.

The plan for the next working session is to make the server (optionally) use the CLI tool to compute matting operations.