Paparazzo
Automated Imaging System

Florian Feigl, BSc.

17th May 2025

Contents

1 Introduction

Paparazzo is a modular, automated imaging system developed for photographing well plates and similar scientific samples. It combines a Raspberry Pi with camera modules, an Arduino-based motor controller, and a custom software suite to automate high-throughput image acquisition.

2 System Overview

The system consists of the following main components:

- Raspberry Pi 4 with a PiCamera v2.1 module.
- Touch display
- Arduino microcontroller connected via USB, controlling a NEMA 17 stepper motor via a TB6600 driver.
- Motorized rail system to move the camera beneath the sample plate.
- Python software for capturing, naming, and storing images based on well position.

3 Installation

3.1 Hardware Setup

1. Connect the display and the camera to the Raspberry Pi.
2. Connect the stepper motors to the TB6600 drivers and then to the Arduino.
3. Power the motor driver with a 12V supply.
4. Ensure the Arduino is connected to the Raspberry Pi via USB.

3.2 Software Setup

1. Flash the Arduino with the provided firmware using the Arduino IDE.
2. Install the required Python packages on the Raspberry Pi:

```
git clone https://github.com/florianfeigl/paparazzo.git
cd paparazzo
chmod +x install.sh
./install.sh
```

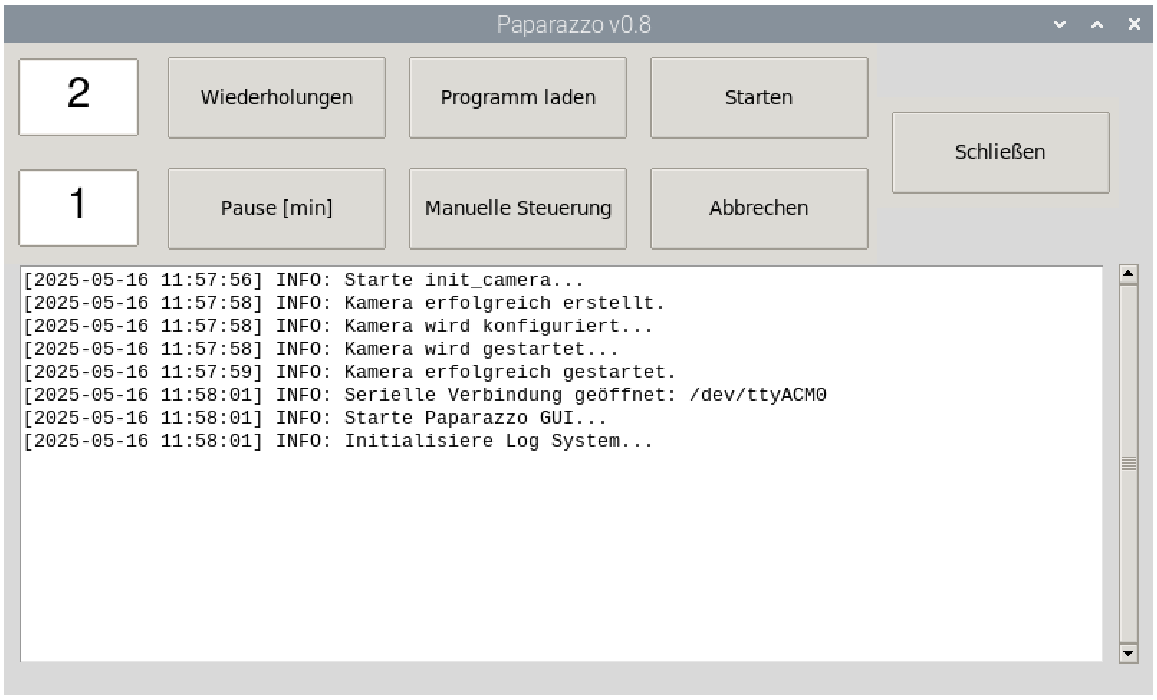
3. Run the paparazzo program: `paparazzo`

4 Usage

The *GUI* is constructed in two lines. *Line 1* consists of four horizontally arranged frames. *Frame 1* contains the configuration panel, where the run properties are set. *Frame 2* holds options to load the firmware and enter a manual mode (not implemented yet). *Frame 3* stores the start and cancel run option. *Frame 4* has the option to close the program/window. *Line 2* offers a detailed real-time log. See Table 1 and Figure 1 and for illustration.

Frame 1: Configure	**Frame 2:** Execute	**Frame 3:** Terminate
Logging: Detailed real-time process information updates.		

Table 1: GUI structure: three operational frames in Line 1, single log area in Line 2



```
[2025-05-16 11:57:56] INFO: Starte init_camera...
[2025-05-16 11:57:58] INFO: Kamera erfolgreich erstellt.
[2025-05-16 11:57:58] INFO: Kamera wird konfiguriert...
[2025-05-16 11:57:58] INFO: Kamera wird gestartet...
[2025-05-16 11:57:59] INFO: Kamera erfolgreich gestartet.
[2025-05-16 11:58:01] INFO: Serielle Verbindung geöffnet: /dev/ttyACM0
[2025-05-16 11:58:01] INFO: Starte Paparazzo GUI...
[2025-05-16 11:58:01] INFO: Initialisiere Log System...
```

Figure 1: Paparazzo GUI v0.8.

4.1 Starting the Imaging Process

1. Launch the control script on the Raspberry Pi.
2. Enter run settings by hitting the according buttons for cycles (`Wiederholungen`) and delay (`Pause [min]`) between cycles, see Figure 2.
3. Load the configurated firmware onto the Arduino using the `Programm laden` button.

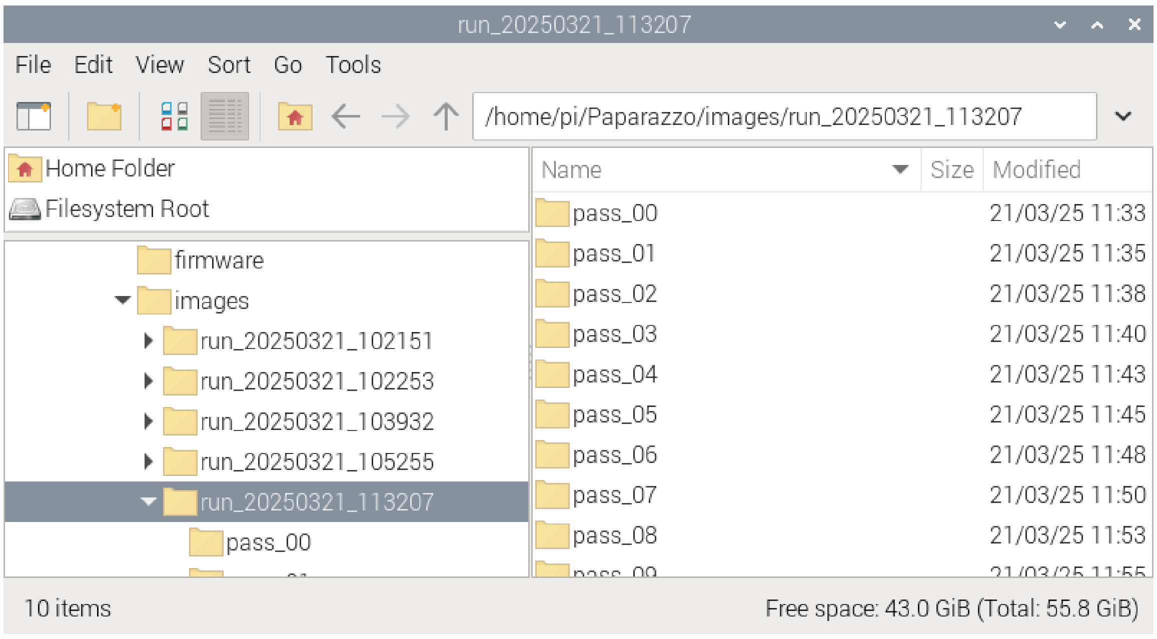
4. Hit the `Starten` button to start the run.

5. The motor will home and then move to each well position in sequence.

6. At each position, an image will be captured with timestamp and named according to the row (A–D) and column (1–6): `{timestamp}_{row_value}{col_value}.jpg`.

4.2 File Output

Captured images are saved in the designated output folder in the following manner (compare Figure 3):
`/home/pi/paparazzo/images/run_{timestamp}/cycle_{CYCLE_COUNT:02d}`



/home/pi/Paparazzo/images/run_20250321_113207

- firmware
- images
  - run_20250321_102151
  - run_20250321_102253
  - run_20250321_103932
  - run_20250321_105255
  - run_20250321_113207
    - pass_00

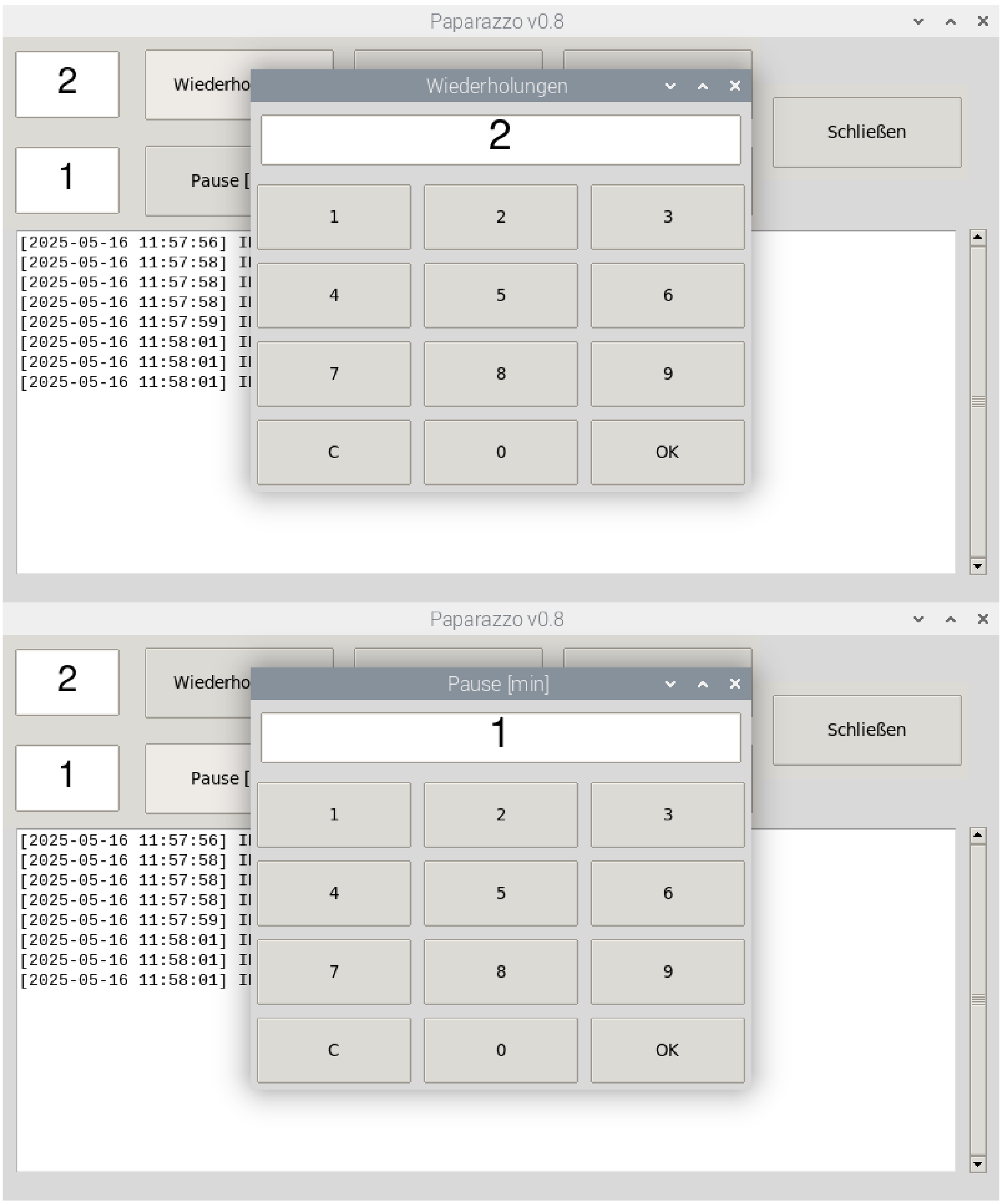
Name	Size	Modified
pass_00		21/03/25 11:33
pass_01		21/03/25 11:35
pass_02		21/03/25 11:38
pass_03		21/03/25 11:40
pass_04		21/03/25 11:43
pass_05		21/03/25 11:45
pass_06		21/03/25 11:48
pass_07		21/03/25 11:50
pass_08		21/03/25 11:53
pass_09		21/03/25 11:55

10 items Free space: 43.0 GiB (Total: 55.8 GiB)

Figure 3: Directory structure for saving data ouput.

5 Maintenance and Troubleshooting

- If you are stuck in an endless reboot loop, try to remove the USB connection to the Arduino and reattach it after booting.
- Ensure the camera lenses are clean before each session.
- Do not manually move elements attached to the motors when they are powered.
- The camera should be aligned as precise as possible beneath the first well, serving as home position.



```
[2025-05-16 11:57:56] I
[2025-05-16 11:57:58] I
[2025-05-16 11:57:58] I
[2025-05-16 11:57:58] I
[2025-05-16 11:57:59] I
[2025-05-16 11:58:01] I
[2025-05-16 11:58:01] I
[2025-05-16 11:58:01] I
```

```
[2025-05-16 11:57:56] I
[2025-05-16 11:57:58] I
[2025-05-16 11:57:58] I
[2025-05-16 11:57:58] I
[2025-05-16 11:57:59] I
[2025-05-16 11:58:01] I
[2025-05-16 11:58:01] I
[2025-05-16 11:58:01] I
```

Figure 2: Configuration of the Paparazzo run.

- If the motor stalls, check power and wiring to the TB6600 and motor. If this occurs, the camera needs to be re-aligned at home position manually.
- If cables loosened, check Figure 4 to ensure correct wiring.
- Serial communication issues can usually be resolved by restarting the script or the Raspberry Pi.

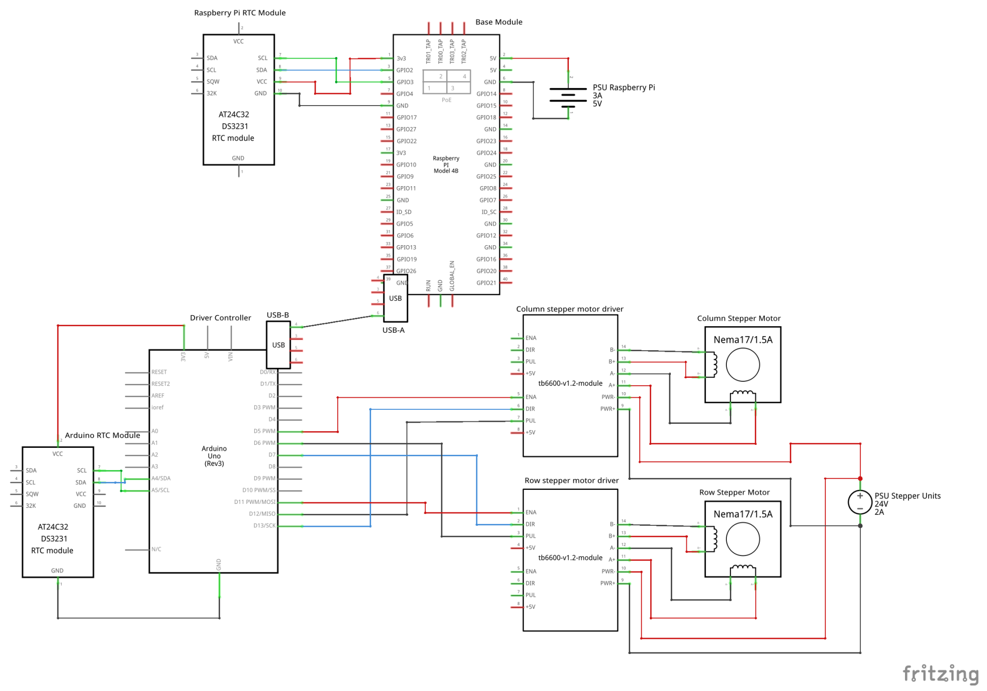


Figure 4: Engineering drawing of the Paparazzo system.

6 Customization

You can adjust:

- Movement parameters in the Arduino sketch (e.g., step delay, microstepping).
- Naming conventions or output format in the Python script.
- Plate layout by changing the grid configuration (currently 4x6 wells).

Note: If you do change anything in the firmware or the python packages, the program needs to be reinstalled (`pip install .`). If you are going to make lots of micro-adjustments, you might consider installing the program in editable install mode, so changes are applied without a reinstall (`pip install -e .`).

7 Contact

For questions or contributions, contact: `florian.feigl@stud.plus.ac.at`