

pyPOCQuant - A tool to automatically quantify Point-Of-Care Tests from images

Andreas P. Cuny^{a,b}, Fabian Rudolf^{a,b}, Aaron Ponti^a

^a*ETH Zurich, Department of Biosystems Science and Engineering, Mattenstr. 26, 4058 Basel, Switzerland*

^b*Swiss Institute of Bioinformatics, Mattenstr. 26, 4058 Basel, Switzerland*

Abstract

Lateral flow Point-Of-Care Tests (POCTs) are a valuable tool for rapidly detecting pathogens and the associated immune response in humans and animals. In the context of the SARS-CoV-2 pandemic, they offer rapid on-site diagnostics and can relieve centralized laboratory testing sites, thus freeing resources that can be focused on especially vulnerable groups. However, visual interpretation of the POCT test lines is subjective, error prone and only qualitative. Here we present pyPOCQuant, an open-source tool implemented in Python 3 that can robustly and reproducibly analyze POCTs from digital images and return an unbiased and quantitative measurement of the POCT test lines.

Keywords: Lateral flow assay (LFA), lateral flow immunoassay (LFIA), Point of care test (POCT), test line quantification, readout zone quantification, diagnostics, Computer vision, QR code, Rapid testing, rapid diagnostic tests (RTD)

1. Motivation and significance

Point of care tests (POCT) and rapid diagnostic tests (RTD) are a cost-effective and instrument-free alternative to molecular laboratory testing used for a wide variety of diseases and the associated immune response, especially in times or regions where resources are limited [1]. Most POCTs are Immunochromatographic Lateral Flow Assays (LFA) or similar, that use one binding protein coated on a membrane and a second protein fused to a colloidal Gold entity to report a positive reaction as a visible color line [2]. The main components of an LFA (Fig. 1a) are, therefore, a nitrocellulose membrane with test and control lines, a sample pad where a conjugate is stored, and the absorption pad [3].

POCTs are commonly used for Streptococcus diagnostics and have been used to combat the epidemics of HIV, TB, hepatitis, dengue fever, and malaria. They all use binding proteins specific for the pathogen as their test line and color conjugate. Analogously, antibody assays reveal whether a patient's immune system has already developed antibodies (i.e., IgM, IgA, or IgG) against past viral infections. A sample is taken using a swab (e.g., in Streptococcus diagnostic), a small amount of blood (e.g., in antibody detection) or similar, and mixed with running buffer either outside or directly on the sample pad of the POCT. In the best case, POCTs deliver fast readouts (typically within 15 minutes) close to the patient and allow for fast reactions with a potential advantage for the patient and the whole local community [4].

To be of use in the global SARS-CoV-2 pandemic, POCTs must fulfill the standards set by the WHO ($\geq 80\%$ sensitivity and $\geq 97 - 99.5\%$ specificity) [5, 6]. These specifications hold both for the diagnostic of an ongoing viral infection and the detection of past infections. A plethora of manufacturers provide tests that they claim to fulfill these specifications, but we argue that they will require comprehensive characterization and comparison of their analytical and clinical performance [7, 8].

LFA and similar devices are scored visually and yield a dichotomous (positive/negative) readout. Experimentally, the analytical limit of detection can be determined using a titration of (repeated) measurements of purified proteins (such as monoclonal antibodies as a surrogate for blood IgG detection) or in proteins from the pathogen of interest that are expressed in cell lines and purified (i.e. Spike protein of SARS-CoV-2) [9]. However, they do have the potential for a quantitative readout using dedicated readers, lab scanner, or image-based analysis [10, 11, 12, 13]. While there have been efforts to use smartphones as readers in combination with 3D-printed adapters and holders [14, 15, 16, 17, 18, 19, 20], no freely available pipeline is available to analyze POCTs from large numbers of images in an automated way.

Here, we present pyPOCQuant, an open-source tool for the large-scale, quantitative and automated analysis of large numbers of POCTs from a variety of different manufacturers independent of design. pyPOCQuant processes images of POCTs taken on digital cameras and returns both a binary result (positive vs. negative) for the test and control lines, and a quantitative readout of each test line. These robust and reproducible readouts are well suited for studying the temporal evolution of the patient's immune response to the infectious agent, or as a quick diagnostics tool for detecting a pathogen. As such, pyPOCQuant helps with the validation and characterisation of different POCTs, but it can also make the use of these assays more reliable.

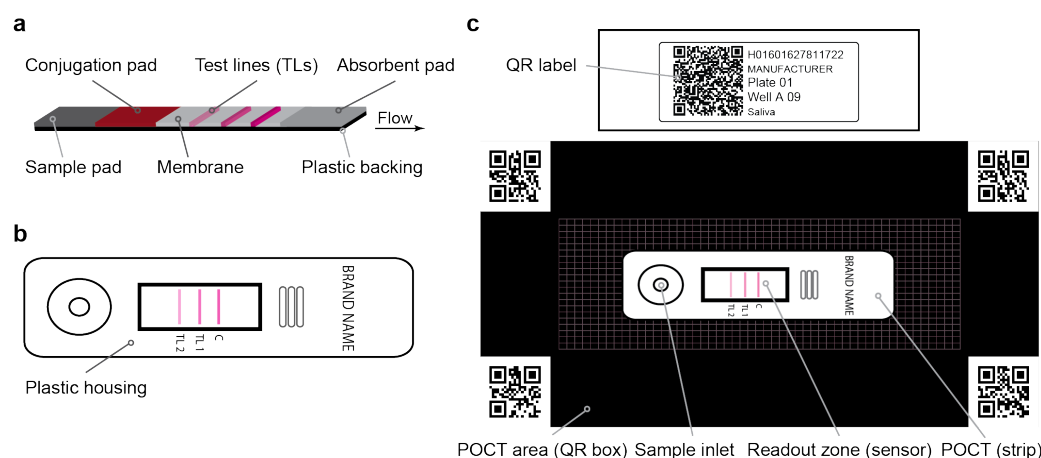


Figure 1: Typical design of lateral flow based POCTs. a) Configuration of LFA. b) POCT schematic enclosing the LFA. c) Example QR code encoding patient metadata (details in text) with POCT area and POCT positioning on template (Suppl. Fig. S1).

2. Software description

pyPOCQuant is a cross platform tool written in Python 3 and implemented as a stand-alone computer vision library. It is available under the open-source license GPLv3 on the Python Package Index (pip) and on GitHub, both as source or as compiled stand-alone executable. Extensive documentation (including example scripts and data) can be found on readthedocs.org and bundled with the application. A quick start guide highlights the steps required to properly set up image acquisition and to run a successful analysis. The user manual describes its usage in full detail.

2.1. Software Architecture

Depending on the workflow, needs and preferences of the user, pyPOCQuant can be launched from the command line interface (CLI) using the `pyPOCQuant.py` main script, or run as a module from a Jupyter notebook. We also developed pyPOCQuantUI, a user-friendly graphical user interface (GUI) and desktop application that leverages pyPOCQuant while massively facilitating the definition of essential parameters needed for the analysis. pyPOCQuantUI is implemented in Python 3 using the PyQt5 library and can be run from source or as a stand-alone, executable desktop application.

Figure 2 displays a flowchart of the operations performed by pyPOCQuant. In short, for every image in a folder, pyPOCQuant extracts patient and metadata information, segments and processes the POCT, extracts its sensor area, locates and quantifies the test-line signals. All results are stored

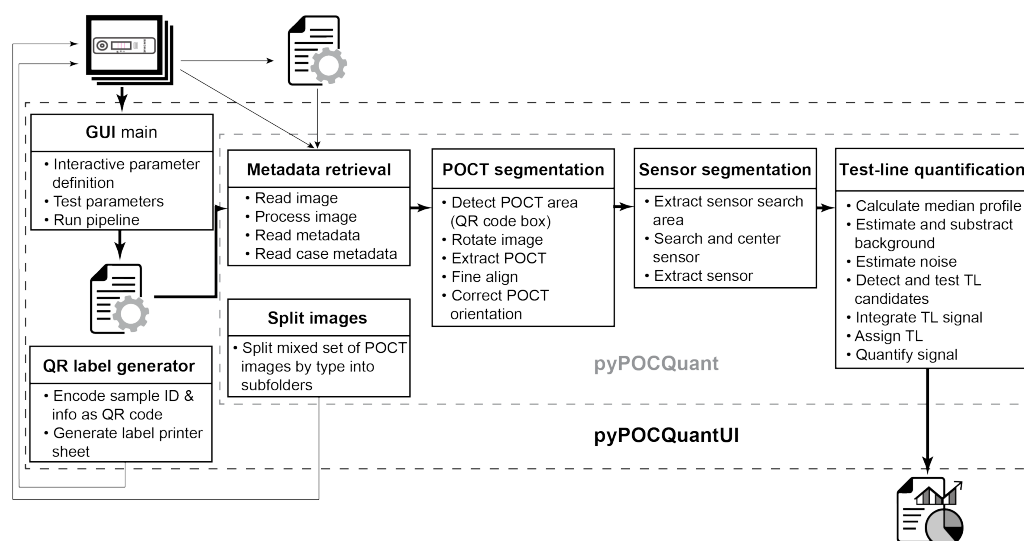


Figure 2: pyPOCQuant architecture overview. Bold arrows depict the main flow through the tool using the GUI. Regular arrows depict the CLI usage and optional ways of interaction.

74 in a data frame that can be used for further statistical analysis. The next
75 sections will discuss these operations in detail.

76 2.2. Software Functionalities

77 The core of pyPOCQuant is a pipeline of operations that can be config-
78 ured and started either from the CLI or GUI. Parameters are passed to the
79 pipeline via a configuration file.

80 2.2.1. Pipeline

81 The pipeline processes all images in a user-defined input folder in par-
82 allel, and the number of parallel tasks can be defined at run time (Figure
83 2). Results are compiled in a common data frame and saved as a comma-
84 separated-value file in a user-defined output folder. The following sections
85 describe the operations that are performed on each image independently.

86 *Metadata retrieval.* Current image from a series of formats (JPEG and var-
87 ious RAW image formats, specifically Nikon NEF, Sony ARW and Canon
88 CR2) is read from disk. Image metadata such as acquisition date and time,
89 exposure time, focal length, ISO, are read and stored in the data frame.
90 Use of the provided template (Suppl. Fig. S1) for image acquisition ensures
91 that patient information stored in a QR code placed on the template can be
92 automatically retrieved and associated directly to the image.

93 *POCT segmentation.* Additional QR codes encoding spatial information make
 94 the process of segmentation more robust. These QR codes ensure that the
 95 image was acquired in the expected orientation. If this is not the case, a
 96 coarse initial rotation (in steps of 90 degrees) is performed. The positional
 97 QR codes delimit the area where the POCT is placed on the template (Suppl.
 98 Fig. S1). The POCT area is large enough to guarantee that POCTs of dif-
 99 ferent shapes and sizes can be contained and robustly identified. The POCT
 100 area is extracted and finely rotated to be parallel to the image axes. The
 101 POCT itself is segmented and rotated to be perfectly aligned to the image
 102 axes. Since it may happen that the POCT was incorrectly rotated by 180
 103 degrees when placed on the template, optional feature detection and OCR
 104 are used to ensure that the inlet and the control TL are on the expected side
 105 of the image.

106 *Readout zone (sensor) segmentation.* Once the properly aligned POCT area
 107 has been segmented from the image, a candidate sensor window of size
 108 `sensor_search_area` and centered at the `sensor_center` coordinates is ex-
 109 tracted. Using the control TL as a visual anchor and its expected position
 110 within the sensor area as defined in `peak_expected_relative_location`, the
 111 search area is properly center around the `sensor_center` coordinates and the
 112 sensor of given `sensor_size` is cropped for further analysis.

113 *Test lines (TL) quantification.* As the first step in the detection and anal-
 114 ysis of the TL signals, a one-dimensional signal profile is calculated as the
 115 median intensity along the short side of the sensor window. This increases
 116 the robustness of the analysis by reducing variations in the TL signals due
 117 to inhomogeneous reactions of the bands (Figure 3). Also, since the edges
 118 of the readout zone often show some impurities, `sensor_border` rows and
 119 columns of the sensor image are ignored when calculating the signal profile.

120 A robust linear regressor (Huber regressor, [21]) is used to fit the back-
 121 ground level, that often shows an increasing or decreasing trend along the
 122 profile. Local minima in the profile are iteratively used to build a background
 123 line that is subtracted from the profile. A small constant subtracted from
 124 the background line guarantees that no values of the subtracted profile are
 125 negative. Next, the profile is low-pass filtered with a small Gaussian ker-
 126 nel ($\sigma = 1$) to clean spurious signal variations, and a local max detector
 127 tuned on the size of the expected TLs provides candidate location for the
 128 test lines. To avoid scoring local maxima due to noisy signal variations, a
 129 thresholding step is performed to make sure that only significant peaks are
 130 considered for further analysis. This threshold is calculated from the me-
 131 dian variation of the background local minima and the multiplicative factor

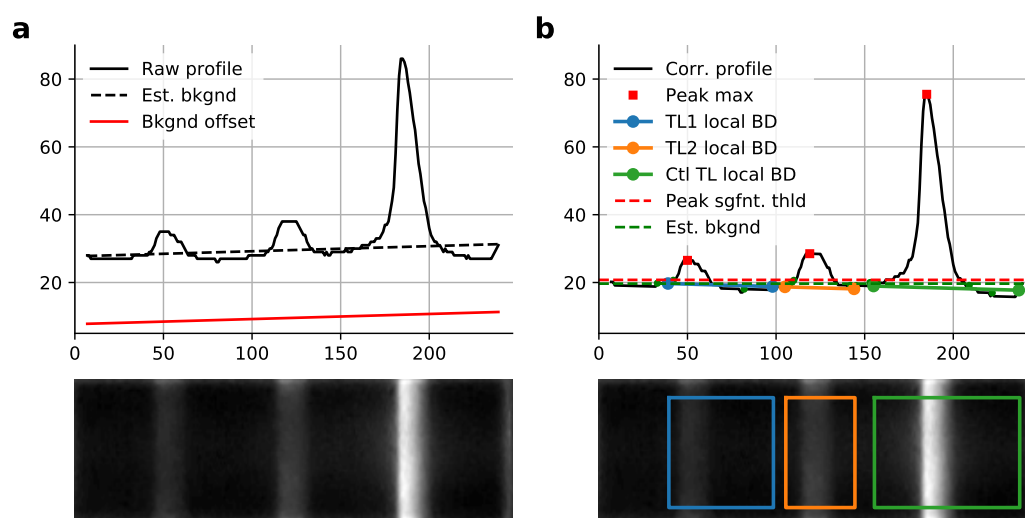


Figure 3: Test line quantification. a) Background estimation (Est. bkgnd) and fixed offset (Bkgnd offset) from raw median intensity profile. b) Background corrected intensity profile (Corr. profile) with peak maxima (red squares) and TL bounds (BD: blue, orange, green circle and lines) as well as peak significance threshold (red dashed line) and the estimated background (black dashed line). The raw grey-scale image plotted below. The colored squares depict all the pixels considered for the signal quantification.

132 **sensor_thresh_factor**. The remaining local maxima are scored by their
 133 relative proximity to the **peak_expected_relative_location** and assigned
 134 to one of the T1, T2 or Ctl TLs. Last, each TL signal is calculated as the
 135 integral of the profile from background level to background level on the two
 136 sides of the peak. This decision was taken after observing that a binary seg-
 137 mentation of the TLs could result in a significant variation in the extent of
 138 the TLs across images, and thus increase the error rate in the detection of
 139 absolute and relative signal strengths.

140 Once all TLs have been extracted and their absolute signals calculated,
 141 the binary readout is recorded. A TL is present if a significant peak could be
 142 extracted and its relative signal strength is normalized against the absolute
 143 signal of the control TL. Absolute values are also reported in the results
 144 data frame to make sure that intra-batch variability can be accounted and
 145 corrected for. If a control TL cannot be found and extracted from the sensor
 146 area, the test is recorded but is labeled as invalid.

147 **Outputs**. The results for each image are collected into a data frame and saved
 148 into the output folder. The structure of the results file is described in the
 149 user manual. Result images and a full log of the process are saved along. The
 150 verbosity of the log and the creation of additional quality control images can

151 be controlled in the settings.

152 2.2.2. Patient sample QR code label generator

153 When working with many POCTs from many patients and/or replicates,
 154 it is essential to link samples and patients unequivocally. Typically, an iden-
 155 tification number is associated with each sample. To facilitate the association
 156 of patient and test metadata with the corresponding POCT analysis read-
 157 outs, we have added a QR-code generation tool to pyPOCQuantUI that al-
 158 lows flexible printing of labels in the format **SAMPLEID-MANUFACTURER-PLATE**
 159 **-WELL-USERDATA**. The label can contain additional information such as sam-
 160 ple id, manufacturer names, identification for sample replicates, organisation
 161 in standardized plates (especially 96-well plates) and a short free text. With
 162 the default page and label settings the full label should not exceed 60 char-
 163 acters. The generated labels are compiled in one PDF file and can be printed
 164 with ordinary printers and placed on the template next to the POCT (Fig.
 165 1c). pyPOCQuant will read this information and add it to the corresponding
 166 analysis results in the data frame.

167 3. Illustrative Examples

168 3.1. Example usage with the GUI

169 pyPOCQuantUI (Figure 4) facilitates the definition of the analysis pa-
 170 rameters by offering interactive tools and visual cues. Also, its immediate
 171 feedback allows checking results and fine-tuning parameters quickly. The
 172 user manual describes all functionalities of the interface in full detail.

173 *Configuration and working folders.* pyPOCQuantUI starts with a default
 174 configuration already loaded. Alternatively, configuration files can be loaded
 175 from the file menu, the toolbar, or simply clicking on a *.conf file in the
 176 input file list. A new project is started by selecting an image folder (1). By
 177 default, the target results folder will target a **pipeline** subfolder in the input
 178 folder, but it can be changed (2). The files contained in the input folder are
 179 listed in (3).

180 *Definition of basic parameters.* Selecting one of the images in the list will
 181 display it on the right (4) and trigger the POCT extraction. Its progress is
 182 shown in (5). The extracted POCT is displayed in (6). By clicking on the
 183 **draw sensor icon** in the toolbar, the user can draw a polygon that connects
 184 the four corners of the readout zone of the POCT (7). This interactive sensor
 185 can be edited and deleted. Mouse wheel zooming or using the zoom icons
 186 facilitate the accurate drawing of the interactive sensor. The sensor polygon

also features three vertical test lines that need to be aligned with the test lines to defined their expected relative position. The vertical lines' position can be changed with the mouse or by editing their values in the parameter tree (8).

Test and run pipeline. At this stage, all the basic parameters are set and can be tested by hitting the **Test parameters** button (9). It is recommended (and activated by default) to enable creation of quality control images and verbose logging during testing. When the parameters deliver satisfying results, one can increase the number of parallel workers (10) and start batch-processing all images in the input folder in parallel (11). The **quantification_data.csv** result file, log files, and, optionally, the quality control images will be stored in the output folder.

Advanced topics. pyPOCQuant's advanced parameters and detailed discussion of the results can be found in the User manual or in the Quick-start guide, which also summarizes a few common issues and suggests ways to resolve them. Worth mentioning is the **Issue** column in the result data frame that contains a non-zero error code if the analysis of the corresponding image could not be completed successfully. The issue codes can be found in the manual.

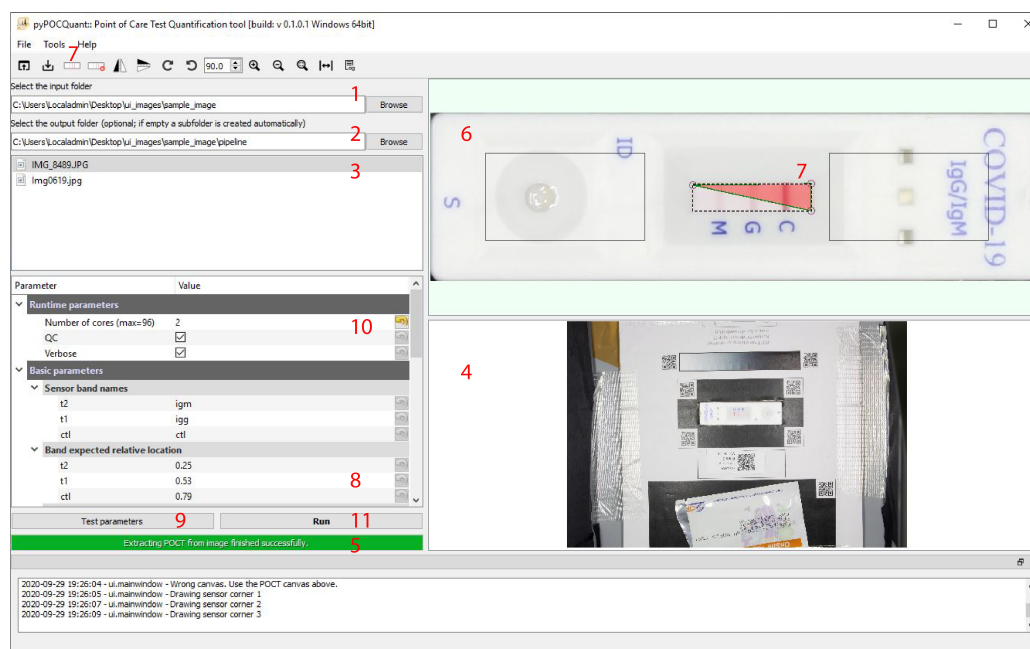


Figure 4: pyPOCQuantUI (GUI): interactively defining basic analysis parameters. The numbers correspond to the text.

206 3.2. Example usage with the CLI

207 Running pyPOCQuant from the CLI is best suited when automating the
208 processing of large amounts of images and folders. A configuration with
209 default values is created (Listing 1) and the parameters are modified appro-
210 priately. The easiest approach is to use pyPOCQuantUI for this purpose,
211 but it could also be done with other tools, such as Fiji (as described in the
212 manual). To create a default configuration from the CLI, use the '-c' flag
213 of pyPOCQuant.py.

Listing 1: Create a new default settings file

```
214 python pyPOCQuant.py -c /PATH/TO/CONFIG/FILE.conf
```

215 Once the configuration file is ready, a full study can be started by run-
216 ning pyPOCQuant on a full folder of images (Listing 2). The analysis is
217 performed in parallel, and the number of concurrent tasks can be defined by
218 the '-w' (--workers) argument. The maximum number of workers allowed
219 corresponds to the maximum number of CPU cores.

Listing 2: Usage of pyPOCQuant with the CLI on an image folder with given settings file and number of workers

```
220 python pyPOCQuant.py -f /PATH/TO/INPUT/FOLDER -o /PATH/  
221 TO/RESULTS/FOLDER -s /PATH/TO/CONFIG/FILE -w ${  
222 NUM_WORKERS}
```

223 The argument '-f' is the path to the folder that contains all images for
224 a given camera and manufacturer to be processed; '-o' is the path where
225 the 'quantification_data.csv' result file, log and quality control images
226 are saved (optional, if omitted the path defaults to a pipeline sub-folder in
227 the input folder); '-s' sets the path to the configuration file to be used for
228 this analysis; '-w' is the number of parallel processes; e.g. 8.

229 3.3. Example usage with Jupyter notebook

230 Finally, pyPOCQuant can be run from a Jupyter notebook. A full demon-
231 stration notebook along with example images can be found in the repository
232 and the online documentation. Documentation of the parameters and out-
233 puts can be found in the User manual. Importing of run_pipeline and
234 default_settings along with the path to images are enough to run pyPOC-
235 Quant from a Jupyter cell (Listing 3).

Listing 3: Minimal example of usage of pyPOCQuant in a Jupyter notebook

```
236 from pypocquant.pipeline import run_pipeline  
237 from pypocquant.lib.settings import default_settings  
238
```

```

239 # Get the default settings
240 settings = default_settings()
241
242 # Change as needed
243 settings["sensor_band_names"] = ('igm', 'igg', 'ctl')
244
245 # Alternatively, load existing settings
246 # from pypocquant.lib.settings import load_settings
247 # settings = load_settings('full/path/to/settings/file.conf')
248
249 # Set final arguments
250 input_folder_path = 'full/path/to/input/folder'
251 results_folder_path = 'full/path/to/results/folder'
252 max_workers = 8
253
254 # Run the pipeline
255 run_pipeline(
256     input_folder_path,
257     results_folder_path,
258     **settings,
259     max_workers=max_workers
260 )

```

261 3.4. Example results

262 We will soon present a detailed, quantitative study of pyPOCQuant in-
263 volving a large sample of images (manuscript in preparation). As an illustra-
264 tion of the performance of pyPOCQuant, we analyzed one image at random
265 from each of 14 manufacturers in a set of 10,000 POCT images [22]. Figure
266 5 shows the extracted TL over the corresponding POCT sensor and area (as
267 delimited by the four QR codes). We then compared the binary readout
268 from pyPOCQuant with the visual detection of two researchers and found
269 an accuracy of 1.00 (n=31 TLs on 14 POCTs). We summarise the results
270 in the table at the bottom right panel of Figure 5. The Jupyter notebook
271 used for the analysis can be found in the **examples** folder of the code reposi-
272 tory. This first result shows that pyPOCQuant can identify TL from various
273 POCT housing designs with high accuracy.

274 4. Impact

275 We developed pyPOCQuant to analyse POCT images as a rapid and reli-
276 able alternative to visual scoring of these devices. While humans can quickly

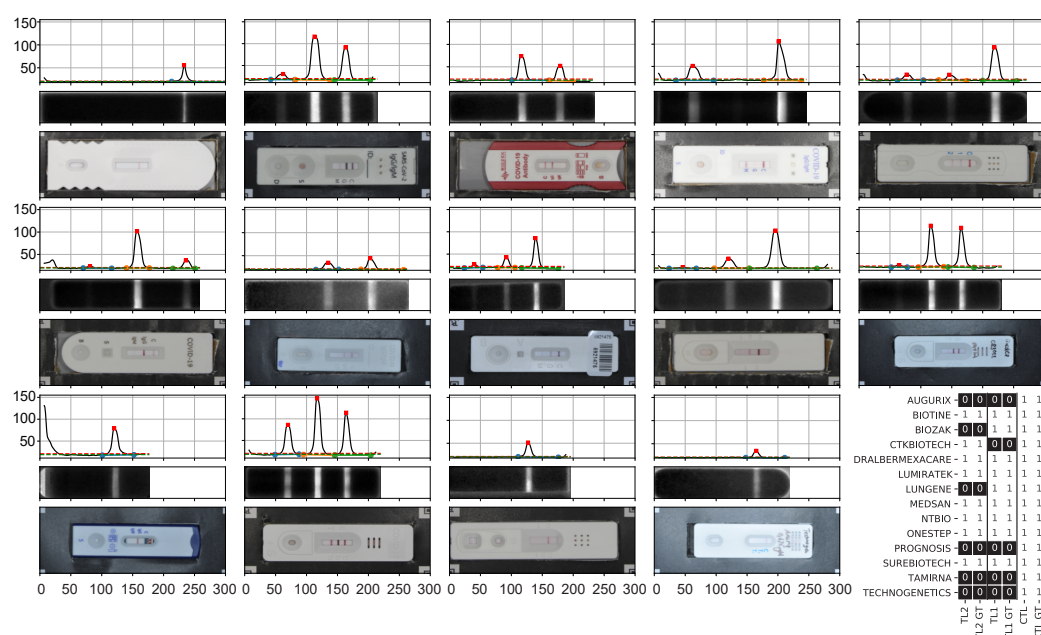


Figure 5: pyPOCQuant results for different POCT manufacturers. Top row depicts the background corrected signal and a red square marks the identified TL. Middle row depicts the detected sensor. Bottom row depicts the POCT inside the raw QR code box. The third entry in the first row shows an example of a POCT that was wrongly positioned in the QR code box and that is ultimately rotated for sensor extraction. The table shows the pyPOCQuant results compared to a manual ground truth for each TL.

and easily spot the presence or absence of TLs of POCTs, this assessment is always subjective and poorly reproducible. Different operators may easily interpret weak test line signals differently and even the same operator may score the same weak test line as positive or negative under different conditions. Therefore, human assessment is neither reliable nor quantitative.

We propose pyPOCQuant as a valuable tool that generates quantitative and reproducible POCT results from lab evaluations all the way to large-scale testing campaigns. pyPOCQuant is not dependent on specialized equipment and can be used with POCTs from many different manufacturers. It therefore allows for straightforward quantification of cheap, mass-producible and readily available POCT using a consumer (or even a smartphone) camera and a mid-end computer.

The efficient scoring of POCTs by pyPOCQuant allows for simple validation, and improves their reliability in every day use. Scaling up is feasible as it can take advantage of multi-core computers to process images in parallel making it suited for large studies with thousands of associated images. As such, pyPOCQuant can help when large-scale testing using classic approaches

such as PCR or ELISA may be hampered by limited access to reagents and laboratory equipment, such as during the global SARS-CoV-2 pandemic.

5. Conclusions

pyPOCQuant delivers accurate detection and precise quantification of test and control lines on any LFA. pyPOCQuant is available as a library, command-line tool and graphical user-interface to fit into various analysis workflows. We developed it to reproducibly batch-process thousands of images of POCTs in parallel using multi-core computers or even clusters. We made pyPOCQuant user-friendly by adding a graphical user interface to facilitate loading images, defining and testing analysis parameters on single images, and launching entire analysis campaigns with a few clicks. In addition to its execution from the CLI or GUI, pyPOCQuant can also run in Jupyter notebooks. Notebooks integrate analysis and documentation into elegant reports.

Image acquisition requirements are undemanding: consumer cameras provide acceptable images for analysis (6-12 MPixel cameras offer a good compromise between image resolution and size), and the provided imaging template makes pyPOCQuant compatible with a wide variety of POCT housing designs and sizes.

pyPOCQuant helps validating POCTs quickly and can be used to establish rapid field testing. It has already been used in several pilot SARS-CoV-2 related projects involving several thousand images. Since POCTs are not restricted to SARS-CoV-2 antigens, pyPOCQuant can be broadly used for large-scale diagnostics and studies of various other diseases.

6. Conflict of Interest

The authors declare no competing interests.

Contribution

Andreas P. Cuny Conceptualization, project administration, methodology, software, resources, investigation, data curation, formal analysis, validation, visualization, original draft preparation, writing, review and editing. **Aaron Ponti** Conceptualization, methodology, software, resources, investigation, data curation, validation, original draft preparation, writing, review and editing. **Fabian Rudolf** Conceptualization, review and editing.

326 Acknowledgements

327 We would like to thank Marvin Wyss and Talea Marty from the Fach-
328 hochschule Nordwestschweiz for valuable user feedback that helped us greatly
329 improve the software and the user experience. We would like to thank Marc
330 Obrist for providing us with a ring light for the photo box during the time
331 of lockdown when resources were generally hard to get. FR is funded by the
332 NCCR 'Molecular Systems Engineering'.

333 References

- 334 [1] C. S. Kosack, A.-L. Page, P. R. Klatser, A guide to aid the selection
335 of diagnostic tests, Bulletin of the World Health Organization 95 (9)
336 (2017) 639.
- 337 [2] K. M. Koczula, A. Gallotta, Lateral flow assays, Essays in biochemistry
338 60 (1) (2016) 111–120.
- 339 [3] B. O. Farrell, C. Immunoassays, Lateral flow immunoassay, Vol. 57,
340 2016. doi:10.1007/978-1-59745-240-3.
- 341 [4] N. P. Pai, C. Vadnais, C. Denking, N. Engel, M. Pai, Point-of-Care
342 Testing for Infectious Diseases: Diversity, Complexity, and Barriers in
343 Low- And Middle-Income Countries, PLoS Medicine 9 (9) (2012). doi:
344 10.1371/journal.pmed.1001306.
- 345 [5] World Health Organization and others, Antigen-detection in the diagno-
346 sis of sars-cov-2 infection using rapid immunoassays: interim guidance,
347 11 september 2020 (2020).
- 348 [6] Centers for Disease Control and Prevention and others, Interim guide-
349 lines for covid-19 antibody testing (2020).
- 350 [7] K. Broich, Antigen tests zum direkten erregernachweis des coronavirus
351 sars-cov-2 (2020).
352 URL <https://antigentest.bfarm.de/ords/antigen/r/antigentes>
353 [ts-auf-sars-cov-2/liste-der-antigentests?session=348348032](https://antigentest.bfarm.de/ords/antigen/r/antigentes)
354 [17141](https://antigentest.bfarm.de/ords/antigen/r/antigentes)
- 355 [8] FDA, Eua authorized serology test performance (2020).
356 URL [http://www.fda.gov/medical-devices/coronavirus-disease-2019-covid-19-emergency-use-authorizations-medical-devic](http://www.fda.gov/medical-devices/coronavirus-disease-2019-covid-19-emergency-use-authorizations-medical-devices/eua-authorized-serology-test-performance)
357 [es/eua-authorized-serology-test-performance](http://www.fda.gov/medical-devices/coronavirus-disease-2019-covid-19-emergency-use-authorizations-medical-devices/eua-authorized-serology-test-performance)
358

- 359 [9] D. A. Armbruster, T. Pry, Limit of blank, limit of detection and limit
360 of quantitation., The Clinical biochemist. Reviews 29 Suppl 1 (August)
361 (2008) S49–52.
- 362 [10] A. E. Urusov, A. V. Zherdev, B. B. Dzantiev, Towards lateral flow
363 quantitative assays: detection approaches, Biosensors 9 (3) (2019) 89.
- 364 [11] C. Minter, A. Casanovas-Massana, T. Li, D. McDonald, L. Wang, S. H.
365 Pan, D. Caianiello, J. Collinski, E. deRamon, R. Hale, et al., Multi-site
366 validation of a sars-cov-2 igg/igm rapid antibody detection kit, medRxiv
367 (2020).
- 368 [12] L. G. Lee, E. S. Nordman, M. D. Johnson, M. F. Oldham, A low-cost,
369 high-performance system for fluorescence lateral flow assays, Biosensors
370 3 (4) (2013) 360–373.
- 371 [13] W. C. Mak, V. Beni, A. P. Turner, Lateral-flow technology: From visual
372 to instrumental, TrAC Trends in Analytical Chemistry 79 (2016) 297–
373 305.
- 374 [14] S. K. Vashist, O. Mudanyali, E. M. Schneider, R. Zengerle, A. Ozcan,
375 Cellphone-based devices for bioanalytical sciences Multiplex Platforms
376 in Diagnostics and Bioanalytics, Analytical and Bioanalytical Chemistry
377 406 (14) (2014) 3263–3277. doi:10.1007/s00216-013-7473-1.
- 378 [15] S. Lee, G. Kim, J. Moon, Development of a Smartphone-based reading
379 system for lateral flow immunoassay, Journal of Nanoscience and Nan-
380 otechnology 14 (11) (2014) 8453–8457. doi:10.1166/jnn.2014.9920.
- 381 [16] T. F. Scherr, S. Gupta, D. W. Wright, F. R. Haselton, Mobile phone
382 imaging and cloud-based analysis for standardized malaria detection and
383 reporting, Scientific Reports 6 (June) (2016) 1–9. doi:10.1038/srep
384 28645.
- 385 [17] S. Dutta, Point of care sensing and biosensing using ambient light sensor
386 of smartphone: Critical review, TrAC - Trends in Analytical Chemistry
387 110 (2019) 393–400. doi:10.1016/j.trac.2018.11.014.
- 388 [18] C. Lee, J. Noh, S. E. O’Neal, A. E. Gonzalez, H. H. Garcia, S. Handali,
389 Feasibility of a point-of-care test based on quantum dots with a mobile
390 phone reader for detection of antibody responses, PLoS Neglected Trop-
391 ical Diseases 13 (10) (2019) 1–11. doi:10.1371/journal.pntd.00077
392 46.

- 393 [19] S. Huang, K. Abe, S. Bennett, T. Liang, P. D. Ladd, L. Yokobe,
394 C. E. Anderson, K. Shah, J. Bishop, M. Purfield, P. C. Kauffman,
395 S. Paul, A. E. Welch, B. Strelitz, K. Follmer, K. Pullar, L. Sanchez-
396 Erebia, E. Gerth-Guyette, G. Domingo, E. Klein, J. A. Englund, E. Fu,
397 P. Yager, Disposable Autonomous Device for Swab-to-Result Diag-
398 nosis of Influenza, *Analytical Chemistry* 89 (11) (2017) 5776–5783.
399 doi:10.1021/acs.analchem.6b04801.
- 400 [20] A. Zhdanov, J. Keefe, L. Franco-Waite, K. R. Konnaiyan, A. Pyayt,
401 Mobile phone based ELISA (MELISA), *Biosensors and Bioelectronics*
402 103 (December 2017) (2018) 138–142. doi:10.1016/j.bios.2017.12
403 .033.
- 404 [21] P. J. Huber, E. M. Ronchetti, *Robust Statistics*, Second Edition, 2009.
405 doi:10.1002/9780470434697.
- 406 [22] F. Rudolf, H.-M. Kaltenbach, J. Linnik, M.-T. Ruf, C. Niederhauser,
407 B. Nickel, D. Gyax, M. Savic, Clinical characterisation of eleven lateral
408 flow assays for detection of covid-19 antibodies in a population, *medRxiv*
409 (2020).

Supplementary material for pyPOCQuant - A tool to automatically quantify Point-Of-Care Tests from images

Andreas P. Cuny^{a,b}, Aaron Ponti^a, Fabian Rudolf^{a,b}

^a*ETH Zurich, Department of Biosystems Science and Engineering, Mattenstr. 26, 4058 Basel, Switzerland*

^b*Swiss Institute of Bioinformatics, Mattenstr. 26, 4058 Basel, Switzerland*

1 *POCT template and mount*

2 To maximize reproducibility in comparing hundreds or thousands of POCT
3 images, we designed a simple mount for the POCT that uses a printable tem-
4 plate in A4 format. The template includes QR codes to assist the localization
5 of the POCT (Suppl. Fig. S1). The top QR code (TL_P) allows to identify
6 a POCT packaging with LOT number. The QR box with the QR codes TL,
7 TR, BL, BR identifies the POCT on the image. The red grid helps to cut
8 out an appropriate window for the POCT to be tested. The QR code R_G
9 and L_G surround grey scale bars. The printed template is glued on a thick
10 carton (approximately 5mm), and the central part is cut to create a pocket
11 that keeps the POCTs stably in position (Suppl. Fig. S2).

12 *Image acquisition setup*

13 Another critical aspect of the acquisition of images for robust analysis is
14 homogeneous lighting conditions with no shadows that can bias the quantifi-
15 cation and a fixed acquisition perspective. The most straightforward setup
16 consists of the POCT mount described in the previous section immobilized
17 on a table and an SLR camera mounted on a tripod. The image is shot in a
18 dark room or a box. A ring light ensures constant, even light illumination;
19 however, a normal flash works as well. For our studies, we built a photo box
20 (Suppl. Fig. S3) from OpenBeam aluminum profiles, a black carton, a ball
21 head mount for the SLR camera (Nikon D90 with a 24mm f2.8 Objective), a
22 Doerr LED Macro Ring Light with 16 LEDs (Doerr LED-16 Ringlight, Art.-
23 Nr. 371023) modified to be wall powered, and a few screws. Additionally,
24 we built a stand from OpenBeam profiles, that allows a smartphone to be
25 placed at a constant height above our template.

26 *Recommendation for improved POCT housing design*

27 POCTs feature high sensitivity and are cost effective, but their format
28 does not lend itself for large-scale, batched analysis. A few dedicated readers
29 exist, but they only work with specific manufacturers and models. pyPOC-
30 Quant relies on a specially-designed template and a configuration file to sig-
31 nificantly extend the number of POCT types that can be analysed at large
32 scale. Still, the process could be simplified even further if the POCT housing
33 were modified to store specific metadata information in machine-readable
34 form. The modifications would be minor. We suggest printing a QR code
35 box around the LFA sensor for robust extraction of the area to quantify and
36 add LOT, date, and possibly additional information as a data matrix directly
37 on the housing. Any camera could read this information, thus making many
38 currently essential computer vision algorithms in pyPOCQuant no longer
39 necessary.

40 *Possible future developments*

41 While the typical use case for pyPOCQuant is during controlled clinical
42 trials of pandemic (such as SARS-CoV-2), it could be deployed in the cloud
43 for use by organizations or state authorities. The operator would interact
44 with the cloud using a web application. The server would produce a unique,
45 random, and anonymous ID that the operator would place on the imaging
46 template during image acquisition. The acquired image would be uploaded
47 and analyzed in the cloud. Results would then be added to a secured database
48 and used to generate live statistics, plots, and reports that could inform the
49 public and drive further actions.

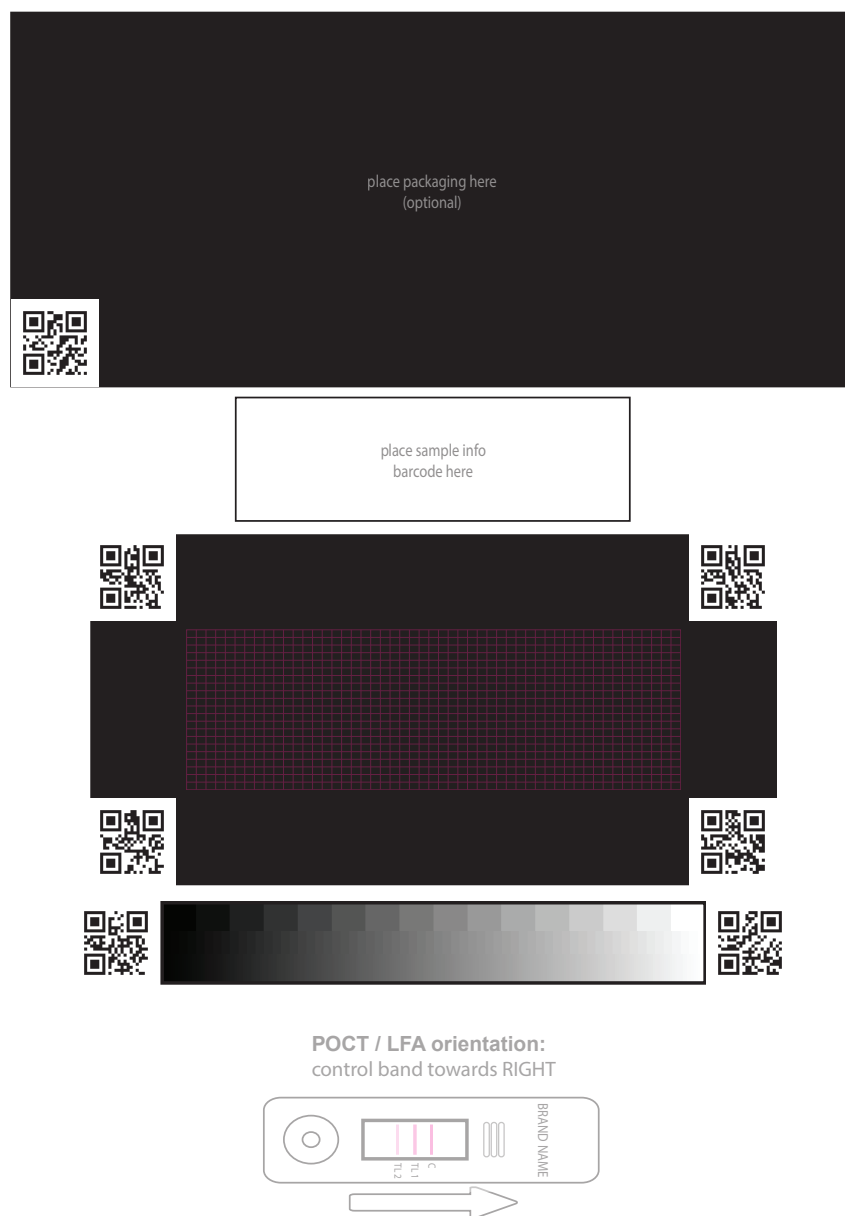


Figure S1: pyPOCQuant POCT recognition template.

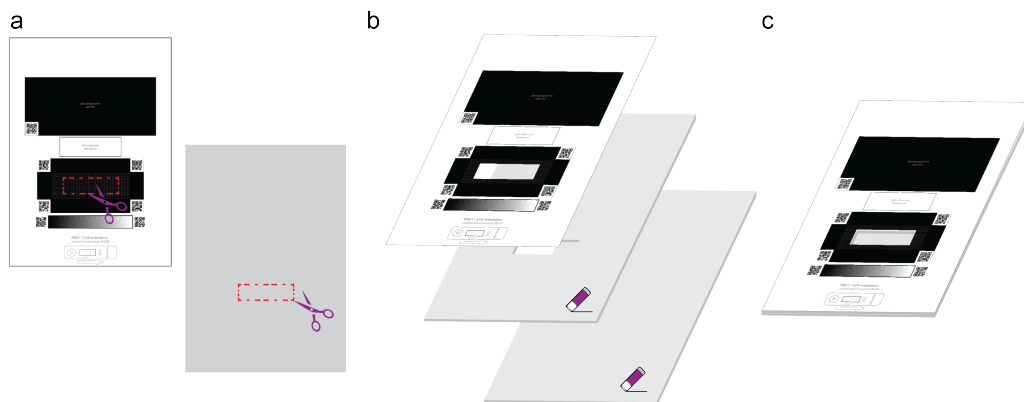


Figure S2: Assembly guide for a simple POCT mount. a) Print the POCT recognition template (Suppl. Fig. S1) in A4 format on a paper and cut out the area for the POCT and do to the same on a carton. c) Glue the template paper on the prepared carton and add a uncut carton below. c) Final mount



Figure S3: Depiction of our image acquisition setup. Top row, OpenBeam smartphone mount above initial version of the POCT template and mount. Bottom row, custom, photo box with controlled light conditions and fixed mounted camera.

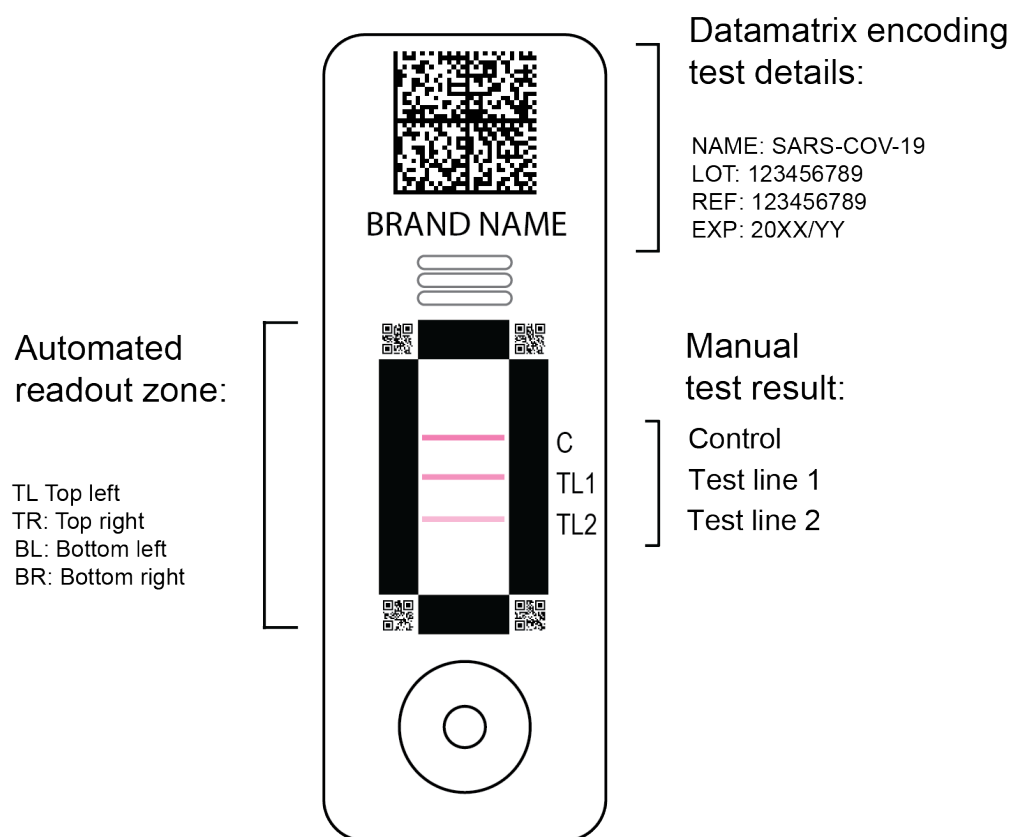


Figure S4: Suggestion for new machine readable POCT housing design.