Date: 24/12/2025

Authors

UNIGE: Maurizio Valle, Lucia Seminara, Christian Gianoglio, Mohamad Yaacoub, Razan Khalifeh.

UPC: Jan Rosell, Raul Suarez, Leopold Avellaneda, Pol Canyameres, Isiah Feihoo.

Feedback on the Tested Force Reconstruction Algorithms

Figure 1 illustrates the results of real-time testing of the force reconstruction algorithm provided by UNIGE in UPC using the Triago robot, with sensor arrays integrated on both clamps. The experiment followed the predefined protocol in which a rigid object was grasped by grasping, held for a short duration, and then released (see APPENDIX A.1. for further details). As shown in Figure 1, a drop in the integral signal occurs in the region highlighted by the dashed black ellipse, immediately before the signal begins to increase again. This drop corresponds to the release phase; however, the algorithm failed to correctly detect this event and, consequently, did not reset the integral signal. This behavior is attributed to the number of samples during the abrupt decrease, approximately 30 samples, while the algorithm's window size was set to 200 samples. As a result, this low number of samples was not sufficient for correct state identification.

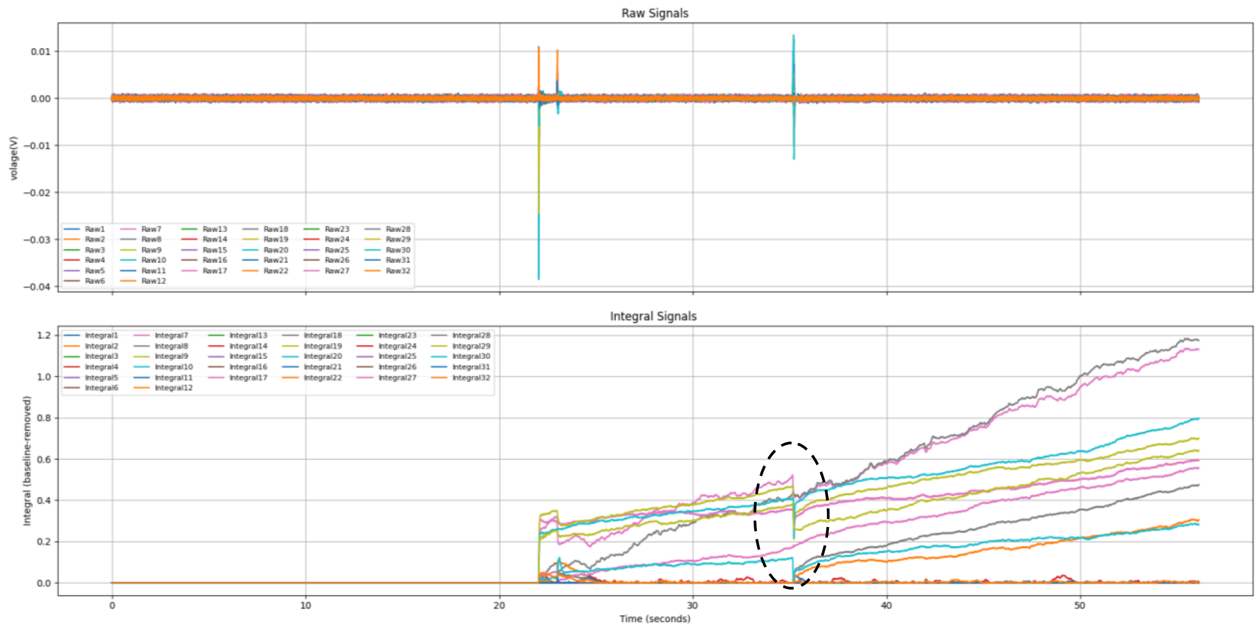


Figure 1. Results of the real-time force reconstruction from UPC.

We investigated this issue on our side and found, as discussed during the meeting, that the problem is mainly attributed to the relatively low transmission signal rate in the system that is currently used by UPC, which is approximately 510 sample/second. As previously pointed out by Pol, the algorithm was not detecting the release event. During the meeting held on 16/12/2025 at 11:00 pm, Mohamad suggested reducing the window size to 100 samples. However, after conducting a systematic analysis to determine the optimal window size (see APPENDIX A.2. to find some examples of the applied windows), we found that an even smaller window is required. Specifically, setting the window size to 20 samples yields significantly better performance.

The results obtained with a 20-sample window are shown in Figure 2. As can be observed, the algorithm more reliably detects the release event and correctly resets the integral signal. **Therefore, we kindly ask you to update the window size in your implementation to 20 samples.**

Additionally, **pol**, after receiving the last detected release window state, we introduced a minor modification to further improve the robustness of the algorithm. Specifically, within the subsequent 20 samples following the release event, the algorithm should verify whether the raw sensor signal is surpassing the thresholds. If the signal does not exceed this threshold, the integral should be reset to zero (algorithm pseudo code version 1.1 in the APPENDIX A.3.).

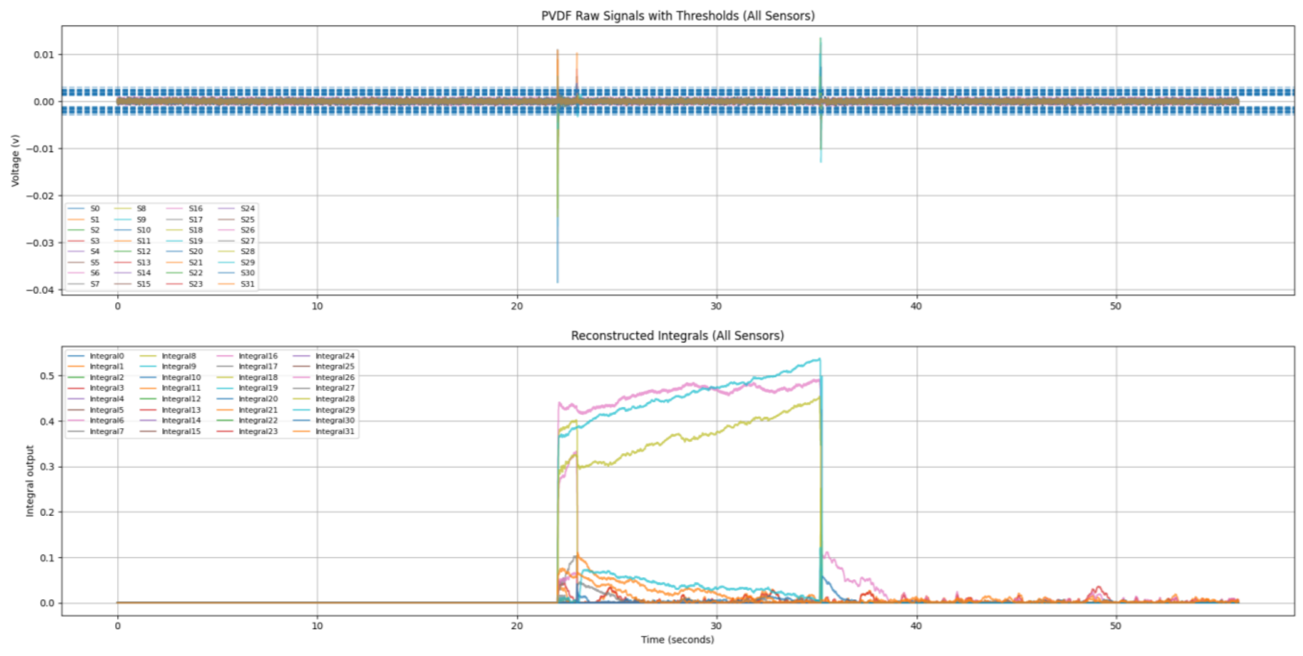


Figure 2. Results of the refined force reconstruction algorithm from UNIGE.

UNIGE requests to UPC

To further strengthen the validation of the proposed algorithm under the new experimental configuration, we kindly request your support with the following activities:

1) Extended Data Collection for Algorithm Validation

We propose collecting an additional dataset to ensure the robustness and reproducibility of the results obtained with the updated configuration (window size = 20).

Experimental protocol:

- Perform a series of trials following the previously established experimental protocol, consisting of grasping a rigid cube, holding it steadily, and subsequently releasing it.
- In contrast to earlier experiments, we kindly request that grasping speed and grasping force be systematically varied. If possible, we recommend using **three levels** for both speed and force (low, medium, and high). Please share the proposed numerical values so that we can review and agree on them before starting the experiment.
- Specifically, we propose:
    - Five sets of trials for each grasping speed across different force levels, where each action lasts for 180 seconds, first waiting for 60 seconds, grasp and hold, and then release for 60 seconds, and then wait for 60 seconds before performing the new grasp
    - Five sets of trials for each force level across different grasping speeds, using the same 160-second trial structure described above.

2) Analyze the source of the unexpected peak

To better understand the origin of the unexpected peak highlighted by the dashed black ellipse in Figure 3. We would greatly appreciate your assistance with the following:

- Please record a video during data collection, as closely as possible with the sensor. This will help determine whether the observed peak arises from:
    - Physical contact with the sensor,
    - An experimental artifact (e.g., unintended contact with the flat wires during data acquisition).
- In parallel, and with your collaboration, we will attempt to identify whether the peak may be attributed to mechanical noise originating from the robot itself.

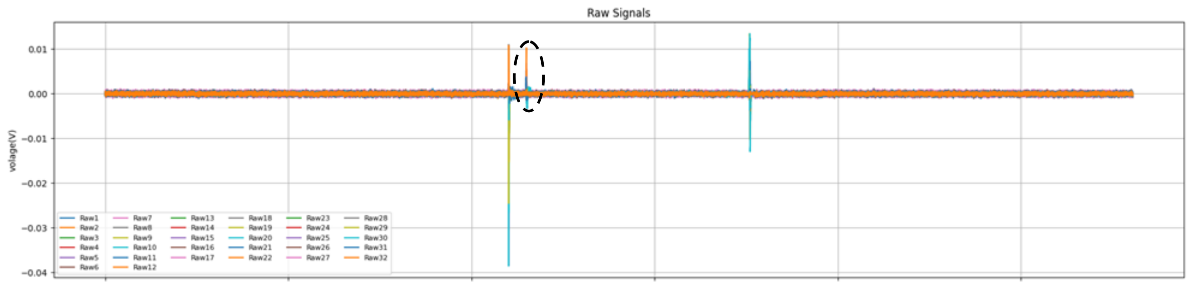


Figure 3 Raw data collected from UPC.

APPENDIX

A.1.

The experimental protocol began by positioning the cubic rigid wooden object on the table and aligning the clamps to grasp before data acquisition from the sensing system, as illustrated in Figure 4.

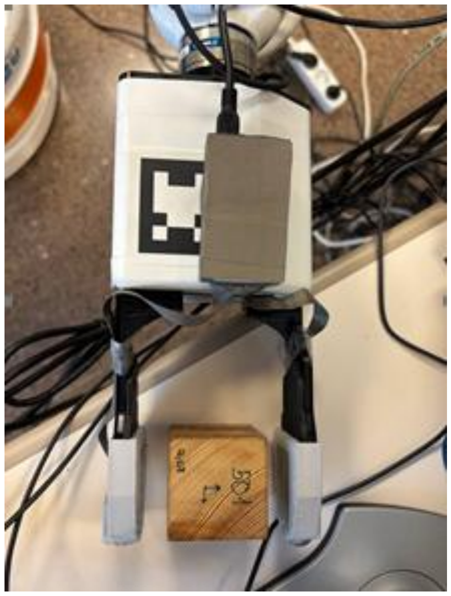


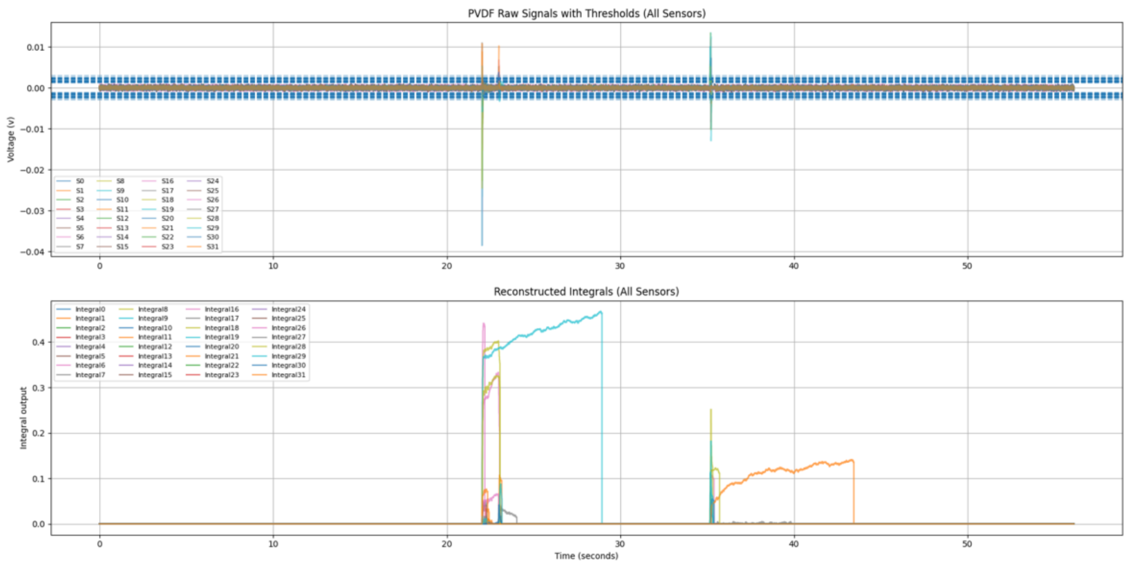
Figure 4 Experimental setup in UPC.

Following the setup, data acquisition was started, with the trial lasting a total of 60 seconds. The procedure comprised three sequential phases: first, a 25-second pre-grasp baseline to define the noise thresholds; second, a 15-second manipulation phase encompassing grasping, holding, and release of the object; and finally, a 20-second post-release before terminating the acquisition.

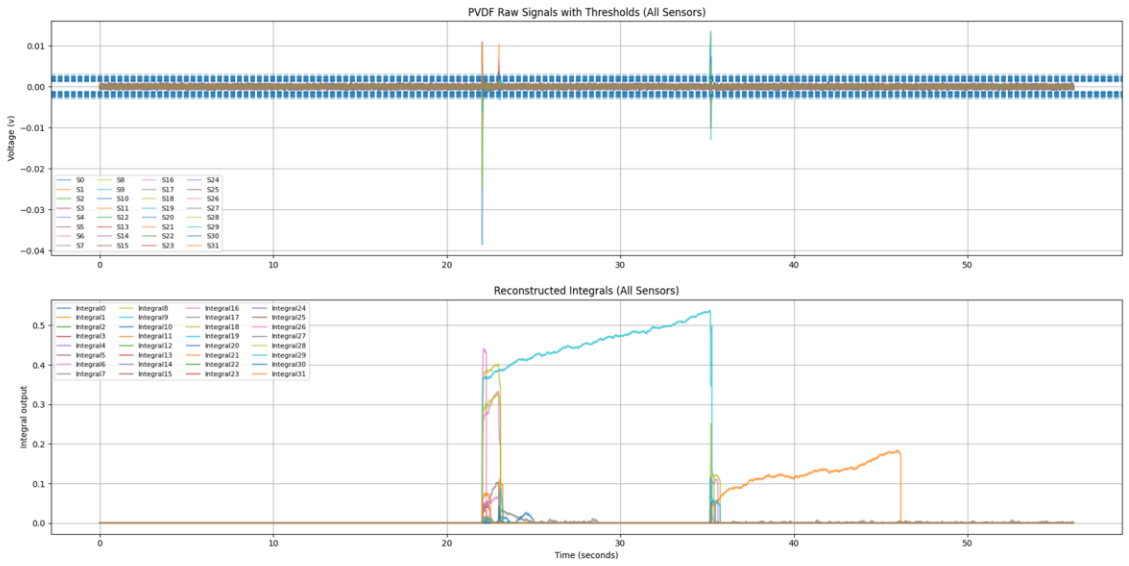
A.2.

Some examples of the results with different window sizes (w)

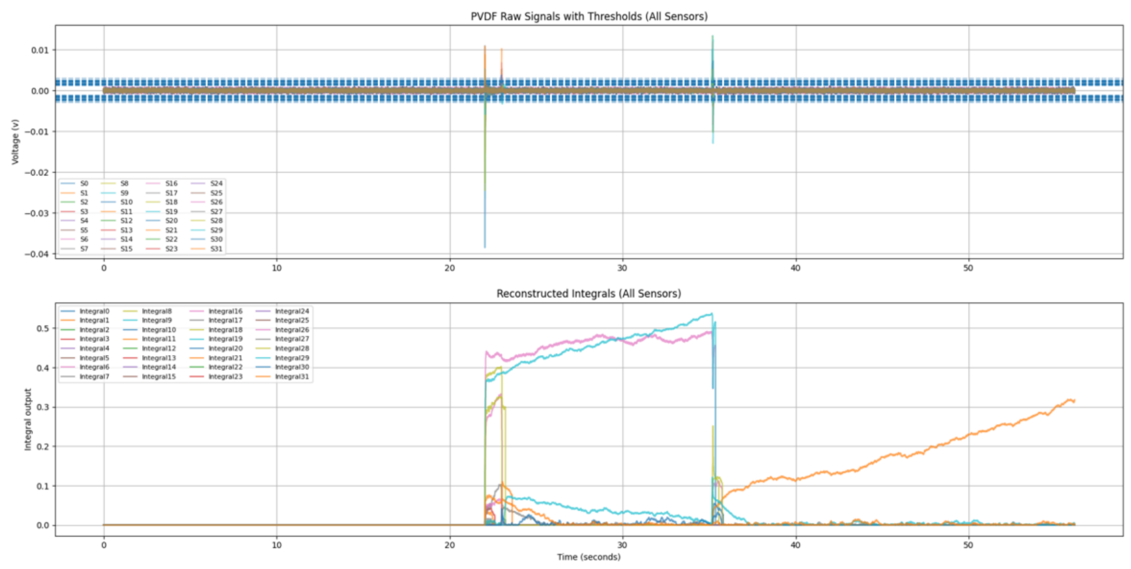
W = 40


PVDF Raw Signals with Thresholds (All Sensors)
Voltage (v)
S0 … S31
Reconstructed Integrals (All Sensors)
Integral output
Integral0 … Integral31
Time (seconds)

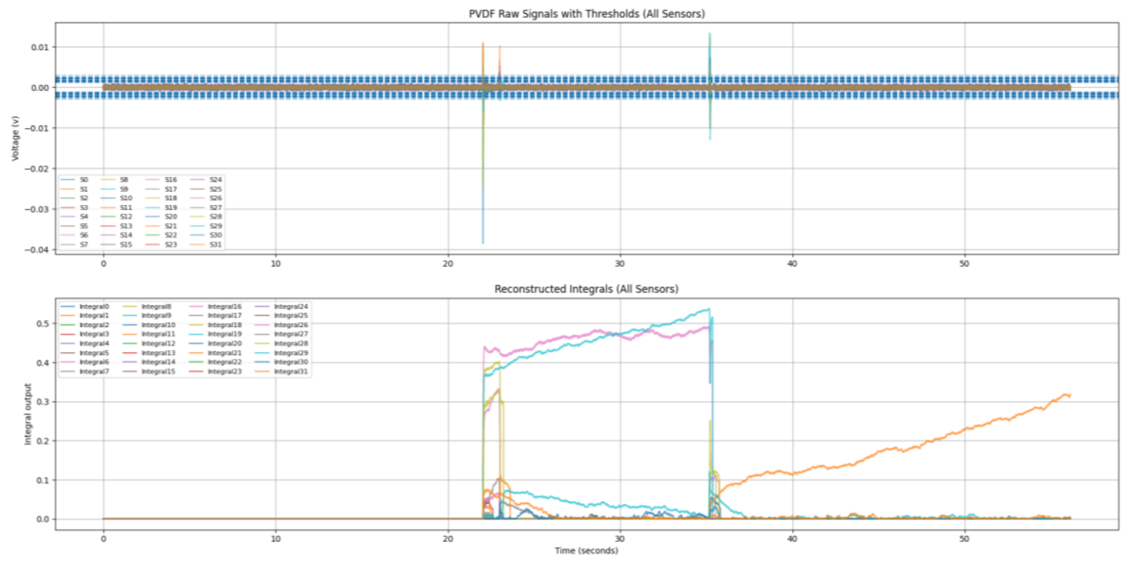
W = 60


PVDF Raw Signals with Thresholds (All Sensors)
Voltage (v)
S0 … S31
Reconstructed Integrals (All Sensors)
Integral output
Integral0 … Integral31
Time (seconds)

W = 80

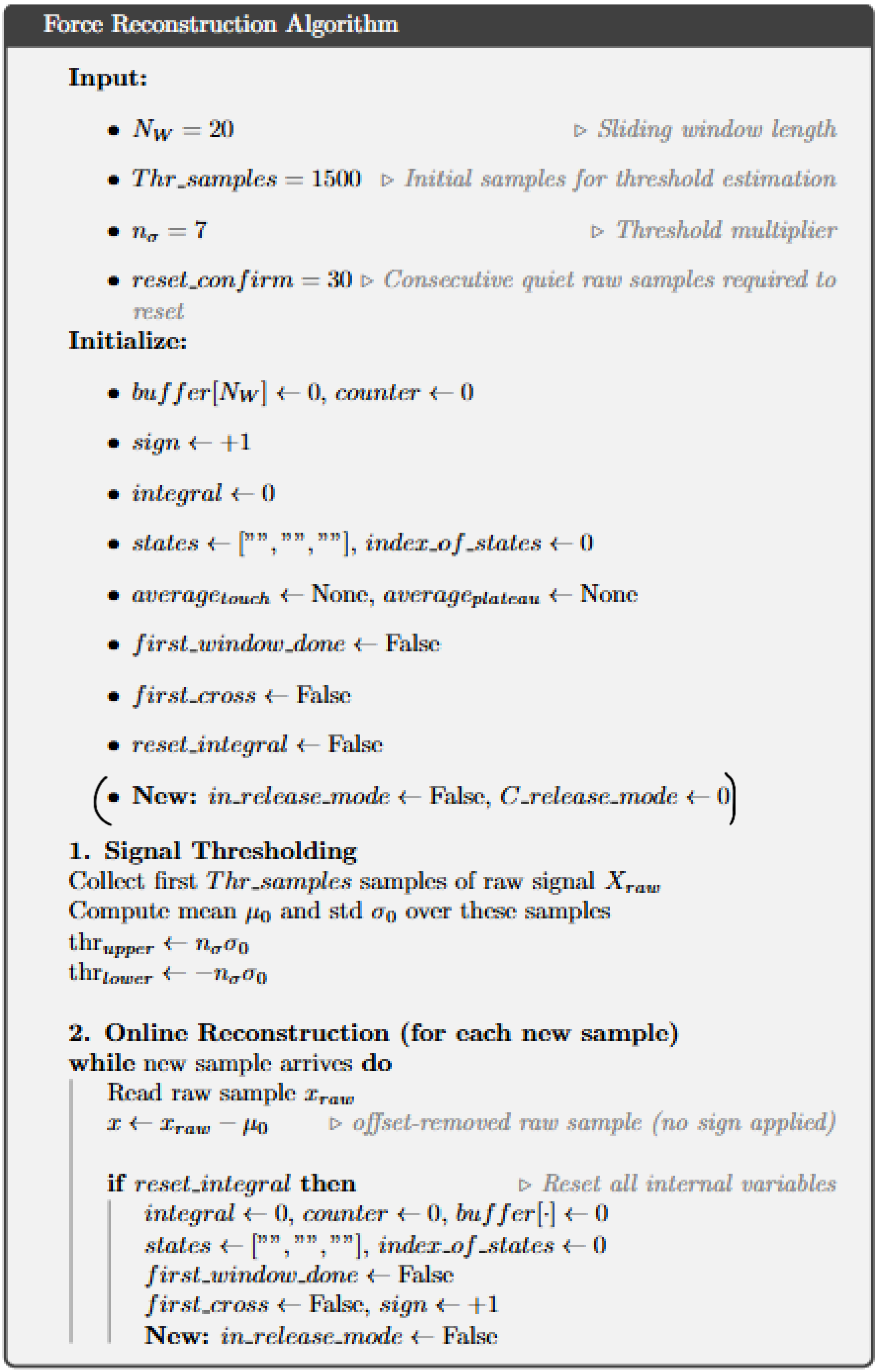

PVDF Raw Signals with Thresholds (All Sensors)
Voltage (v)
Reconstructed Integrals (All Sensors)
Integral output
Time (seconds)

W = 100


PVDF Raw Signals with Thresholds (All Sensors)
Voltage (v)
Reconstructed Integrals (All Sensors)
Integral output
Time (seconds)

A.3.

Version 1.1 of the force reconstruction algorithm



```
Force Reconstruction Algorithm
Input:
  • $N_W = 20$                                    ▷ Sliding window length
  • $Thr\_samples = 1500$   ▷ Initial samples for threshold estimation
  • $n_\sigma = 7$                                ▷ Threshold multiplier
  • $reset\_confirm = 30$ ▷ Consecutive quiet raw samples required to
    reset
Initialize:
  • $buffer[N_W] \leftarrow 0$, $counter \leftarrow 0$
  • $sign \leftarrow +1$
  • $integral \leftarrow 0$
  • $states \leftarrow ["", "", ""]$, $index\_of\_states \leftarrow 0$
  • $average_{touch} \leftarrow$ None, $average_{plateau} \leftarrow$ None
  • $first\_window\_done \leftarrow$ False
  • $first\_cross \leftarrow$ False
  • $reset\_integral \leftarrow$ False
 (• New: $in\_release\_mode \leftarrow$ False, $C\_release\_mode \leftarrow 0$)
1. Signal Thresholding
Collect first $Thr\_samples$ samples of raw signal $X_{raw}$
Compute mean $\mu_0$ and std $\sigma_0$ over these samples
$\text{thr}_{upper} \leftarrow n_\sigma \sigma_0$
$\text{thr}_{lower} \leftarrow -n_\sigma \sigma_0$

2. Online Reconstruction (for each new sample)
while new sample arrives do
    Read raw sample $x_{raw}$
    $x \leftarrow x_{raw} - \mu_0$        ▷ offset-removed raw sample (no sign applied)

    if $reset\_integral$ then                  ▷ Reset all internal variables
        $integral \leftarrow 0$, $counter \leftarrow 0$, $buffer[\cdot] \leftarrow 0$
        $states \leftarrow ["", "", ""]$, $index\_of\_states \leftarrow 0$
        $first\_window\_done \leftarrow$ False
        $first\_cross \leftarrow$ False, $sign \leftarrow +1$
        New: $in\_release\_mode \leftarrow$ False
        $reset\_integral \leftarrow$ False
        **continue**

    **2.1 First threshold crossing detection**
    **if not** $first\_cross$ **then**
        **if** $x > \mathrm{thr}_{upper}$ **or** $x < \mathrm{thr}_{lower}$ **then**
            $first\_cross \leftarrow$ True
            $sign \leftarrow \begin{cases} -1, & \text{if } x < \mathrm{thr}_{lower} \\ +1, & \text{otherwise} \end{cases}$
        **else**                          ▷ *No event yet: keep integral at zero*
            $integral \leftarrow 0$
            $counter \leftarrow 0$
            **continue**

    **2.2 Integration after first crossing**
    $x_{int} \leftarrow sign \cdot x$
    $integral \leftarrow integral + x_{int}$
    **if** $integral < 0$ **then**
        $integral \leftarrow 0$
    $buffer[counter] \leftarrow integral$
    $counter \leftarrow counter + 1$

    **2.3 Confirmation for reset (only in release mode)**
    **if** $in\_release\_mode$ **then**
        **if** $\mathrm{thr}_{lowerr} < X < \mathrm{thr}_{upper}$ **then**
            $C\_release\_mode \leftarrow C\_release\_mode + 1$
        **else**
            $C\_release\_mode \leftarrow 0$
        **if** $C\_release\_mode \geq reset\_confirm$ **then**
            $reset\_integral \leftarrow$ True
            **continue**

    **2.4 Window processing and state labeling**
    **if** $counter == N_W$ **then**
        $avg \leftarrow \dfrac{buffer[N_W - 1] - buffer[0]}{N_W}$
        $counter \leftarrow 0$
        **if not** $first\_window\_done$ **then** ▷ *Initialize reference slopes
        from the first post-touch window*
            **if** $average_{touch}$ is None **then**
                $a \leftarrow \max(|avg|,\ 10^{-12})$
                $average_{touch} \leftarrow 10^{\lfloor \log_{10}(a) \rfloor}$
                $average_{plateau} \leftarrow average_{touch}/10$
            $first\_window\_done \leftarrow$ True
            $average_{plateau} \leftarrow average_{touch}/10$
          $first\_window\_done \leftarrow$ True
          $states[0] \leftarrow set\_label(avg, average_{touch}, average_{plateau})$
        else
          $st \leftarrow set\_label(avg, average_{touch}, average_{plateau})$
                                                    ▷ Update 3-state history
          if $index\_of\_states < 2$ then
            $index\_of\_states \leftarrow index\_of\_states + 1$
            $states[index\_of\_states] \leftarrow st$
          else
            $states[0] \leftarrow states[1]$
            $states[1] \leftarrow states[2]$
            $states[2] \leftarrow st$
        ▷ Enter release mode as soon as we label a window as "release"
          if $st ==$ "release" then
            $in\_release\_mode \leftarrow$ True
             ▷ IMPORTANT: no immediate reset from labels anymore▷ (
tional alternative: if check_states(states) then in_release_mode ←
True)


function SET_LABEL(avg, $average_{touch}$, $average_{plateau}$)
    if $avg \geq average_{touch}$ then
        return "press"
    else if $avg \leq -average_{touch}$ then
        return "release"
    else if $|avg| \leq average_{plateau}$ then
        return "plateau"
    else
        return "transition"

function CHECK_STATES(states) ▷ Original pattern detector (kept for
reference/optional gating)
    if (states[0] == "release" and states[1] == "release") and
    (states[2] == "release" or states[2] == "transition" or states[2]
    == "plateau") then
        return True
    else
        return False
```