## 1 Summary

1€ filter: a tunable algorithm for filtering out high-frequency noise (jitter) from signals sampled in real time.

## 2 Jitter & Lag

When the *precision* of a signal is reduced, one might observe jitter in repeated observations: where values are close but usually not quite right. To remove the unwanted components of a signal, one can filter the signal, but this introduces a decline in *responsiveness* (i.e. as lag – bad for real-time systems!).

We would like to maximize both precision and responsiveness, i.e. minimize both jitter and lag. There is a balance between these two things; the 1€ filter allows users to tune this balance.

## 3 1€ Filter

The 1€ filter utilizes an adaptive low-pass filter, where the cutoff frequency for each new sample ("cut off frequencies above this value") depends on the signal's speed (derivative value). At low speeds, the cutoff is low (less jitter, more lag); at high speeds, the cutoff is high (more jitter, less lag).

Let the cutoff frequency be denoted $f_c$. There are two parameters, $f_{c_{min}}$ and $\beta$.

$$f_c = f_{c_{min}} + \beta|\dot{\hat{X}}_i|$$

where $\hat{X}_i$, the filtered data, is computed from the raw data $X_i$, the time constant $\tau$, the sampling period $T_e$, the filtered previous sample $\hat{X}_{i-1}$, and the smoothing factor $\alpha$ as

$$\hat{X}_i = \alpha X_i + (1-\alpha)\hat{X}_{i-1}$$

**This is the actual filtering.** Note that

$$\tau = \frac{1}{2\pi f_c}$$
$$\alpha = \frac{1}{1+\frac{\tau}{T_e}}$$

and $T_e = \frac{1}{\text{sampling rate}}$ can be computed from timestamps ($\texttt{timestamp} - \texttt{prevTimestamp}$).

The derivative $\dot{\hat{X}}_i$ is what we refer to as the signal speed. We low-pass filter this too.

$$\texttt{speed} = (X_i - X_{i-1}) \cdot \text{sampling rate}$$
$$\dot{\hat{X}}_i\ (\texttt{speed}\ \text{filtered}) = \alpha_s \cdot \texttt{speed} + (1-\alpha_s)\dot{\hat{X}}_{i-1}$$

$\alpha_s$ is based on a separate $f_c$ cutoff frequency for the derivative.

"If high speed lag is a problem, increase $\beta$; if slow speed jitter is a problem, decrease $f_{c_{min}}$."

## 4 1€ Filter: Improved Explanation

After having *used* the filter, here's a refined explanation of things.

The 1€ filter is

$$\hat{X}_i = \alpha X_i + (1-\alpha)\hat{X}_{i-1}$$

It's an exponential moving average; $\alpha \in (0,1)$, so the filtering will go something like

$$\text{filtered value} = \frac{1}{10}\text{ new} + \frac{9}{10}\text{ old}$$

Extremely simple.

- When $\alpha \approx 1$, there's essentially no filtering (max jitter, min lag; return the noisy input as-is).
- When $\alpha \approx 0$, there's essentially no motion (max lag, min jitter; return the initial value every time).

$\alpha$ is defined as

$$\alpha = \frac{2\pi f_c}{2\pi f_c + \text{sampling frequency}}$$

The sampling frequency is $\frac{1}{\text{sampling period}}$, e.g. something like

$$\frac{1}{5.05 - 5.01} = 25$$

if the current value was sampled at time 5.05s and the previous value was sampled at time 5.01s.

$f_c$ is the cutoff frequency, and is defined as

$$f_c = f_{c_{min}} + \beta|\dot{\hat{X}}_i|$$

meaning $\alpha$ is really

$$\alpha = \frac{2\pi(f_{c_{min}} + \beta|\dot{\hat{X}}_i|)}{2\pi(f_{c_{min}} + \beta|\dot{\hat{X}}_i|) + \text{sampling frequency}}$$

where the scary-looking $\dot{\hat{X}}_i$ is really just a filtered derivative, i.e. a filtered

$$\text{change per unit time} = \frac{X_i - X_{i-1}}{\text{sampling period}}$$

They call this the "speed," which makes sense if you think of each value $X_i$ as representing a position.

As speed gets higher

$\rightarrow$ the proportion of $2\pi f_c$ in $(2\pi f_c + \text{sampling frequency})$ gets higher

$\rightarrow$ $\alpha$ gets higher (meaning less lag, more jitter: filtered values more like raw values)

Likewise, as speed decreases $\alpha$ gets lower (meaning less jitter, more lag). This is the *adaptive* nature of the filter: if the value seems to be changing a lot, the filter will increase $\alpha$ for less lag and more jitter (the idea being that if the motion is fast, it would be bad to fall behind).

Increasing $f_{c_{min}}$ and $\beta$ (the tunable parameters) will also increase $\alpha$. The difference between the two is that $f_{c_{min}}$ describes a base value (*always* higher by this much), while $\beta$ describes how $\alpha$ scales with speed.

- Lower $f_{c_{min}}$ and $\beta$ (/lower $\alpha$) $\rightarrow$ slower/laggier but smoother/less jittery

## References

[1] Géry Casiez, Nicolas Roussel, Daniel Vogel. 1€ Filter: A Simple Speed-based Low-pass Filter for Noisy Input in Interactive Systems. CHI 2012.