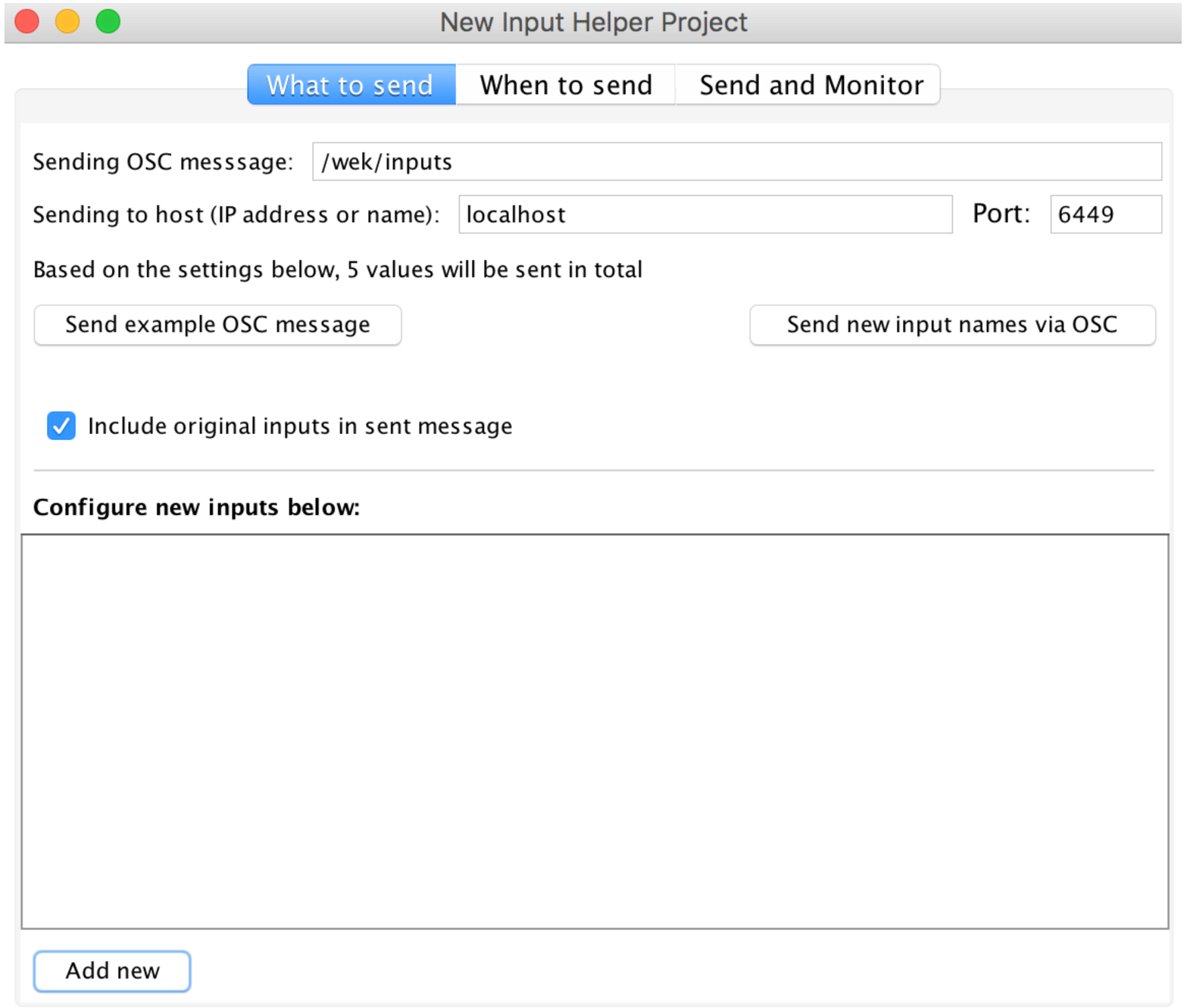
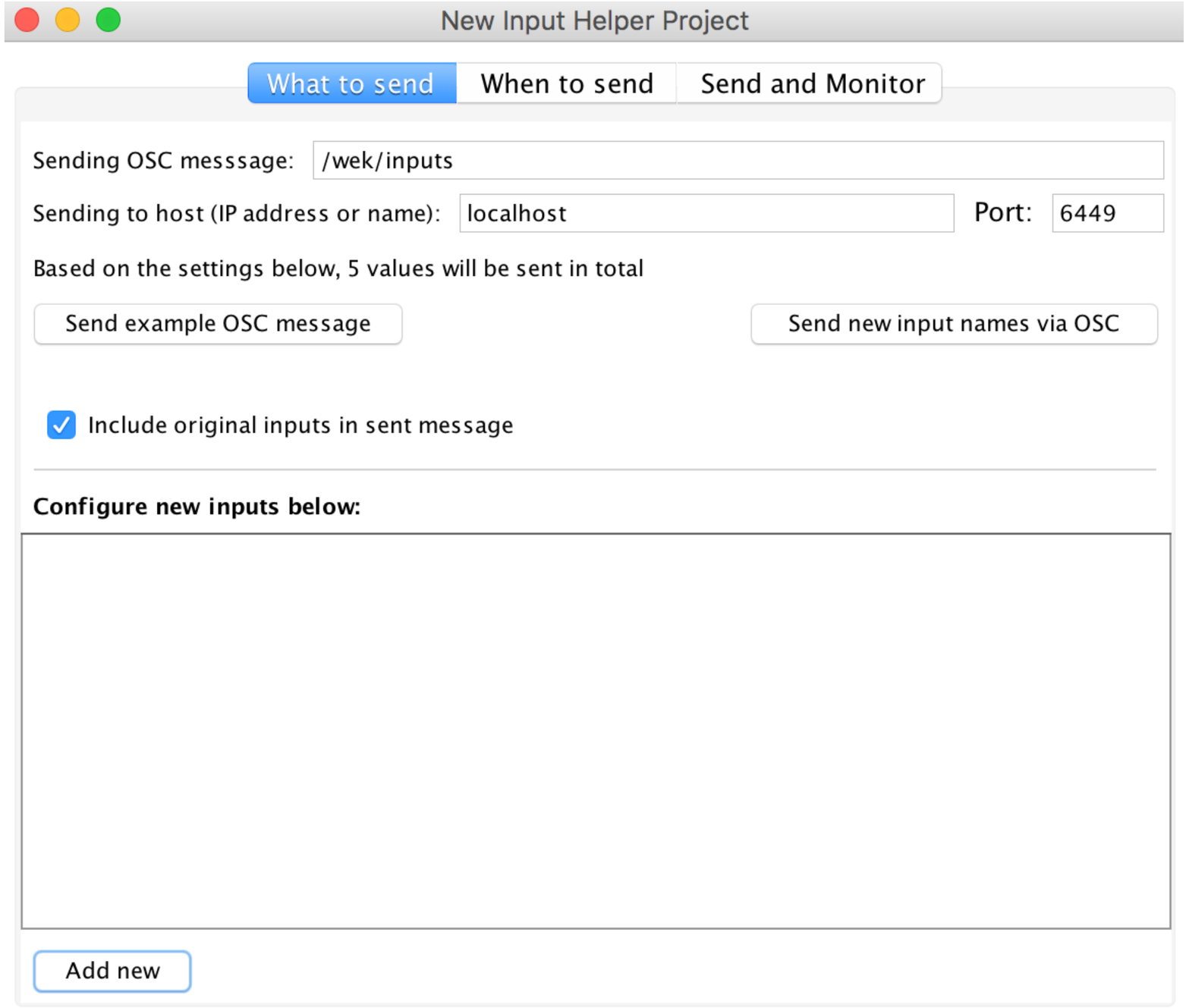
A feature is a measurement of some phenomenon represented by a number.

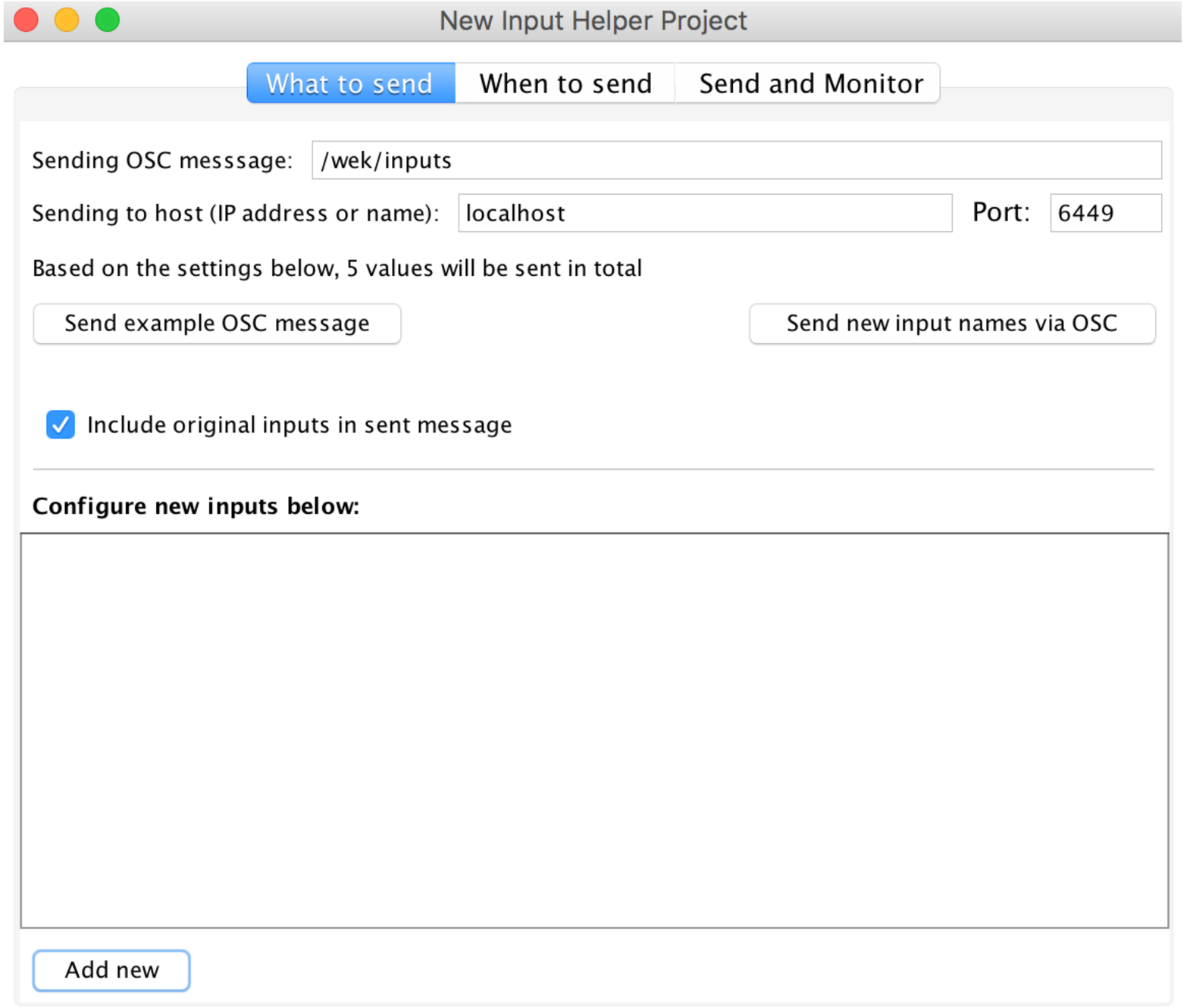
- Only use relevant features
- Have as little as noise possible
- Don’t use too many features
    - (more features mean more training time)
- Features should provide enough info though

Use WekiInputHelper (http://www.wekinator.org/input-helper/) to manipulate inputs in different ways.



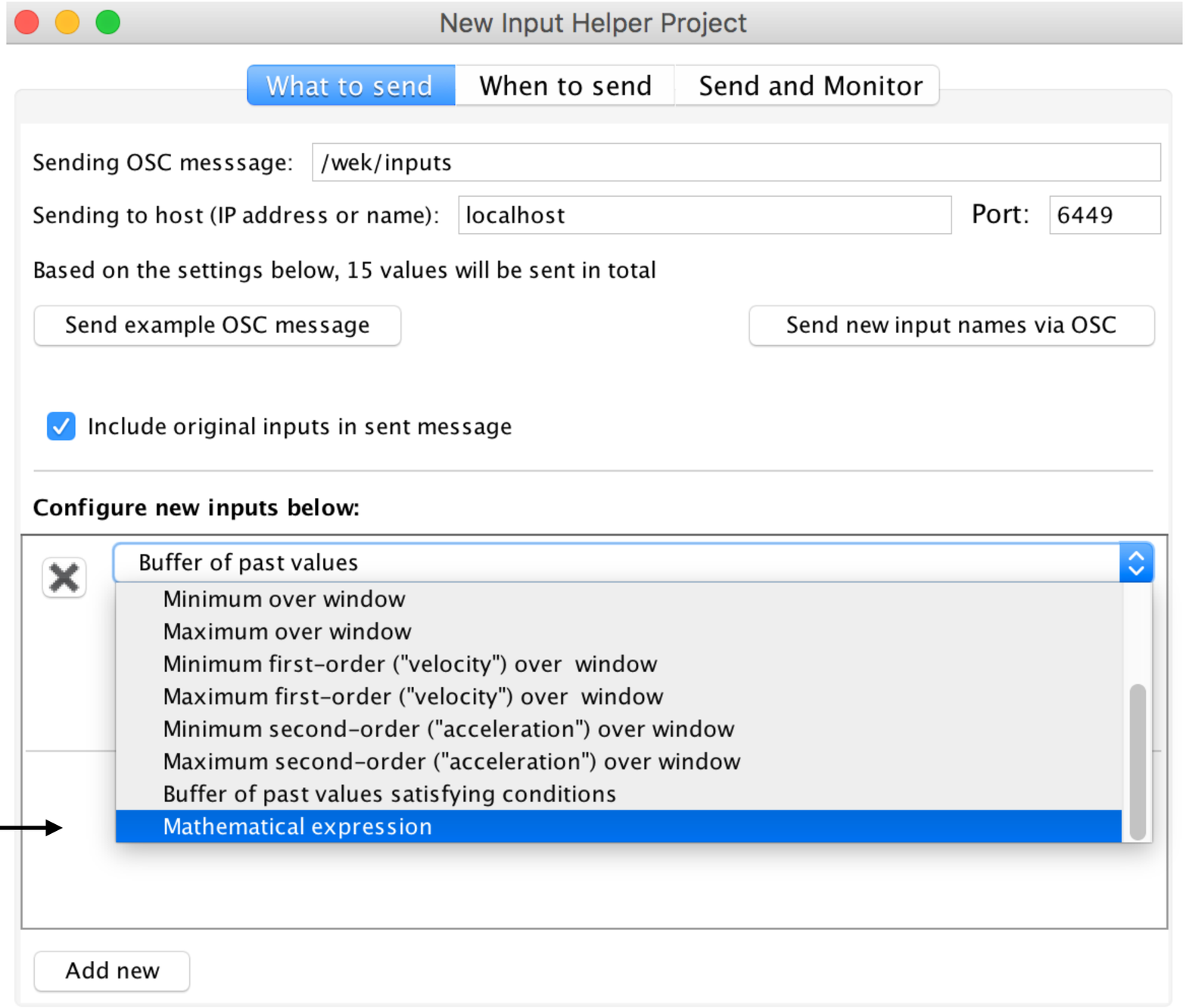
WekiInputHelper sits in-between your input and Wekinator. It will receive on port 6448, and send out to port 6449. You have to change Wekinator to listen to port 6449, instead of the default 6448.





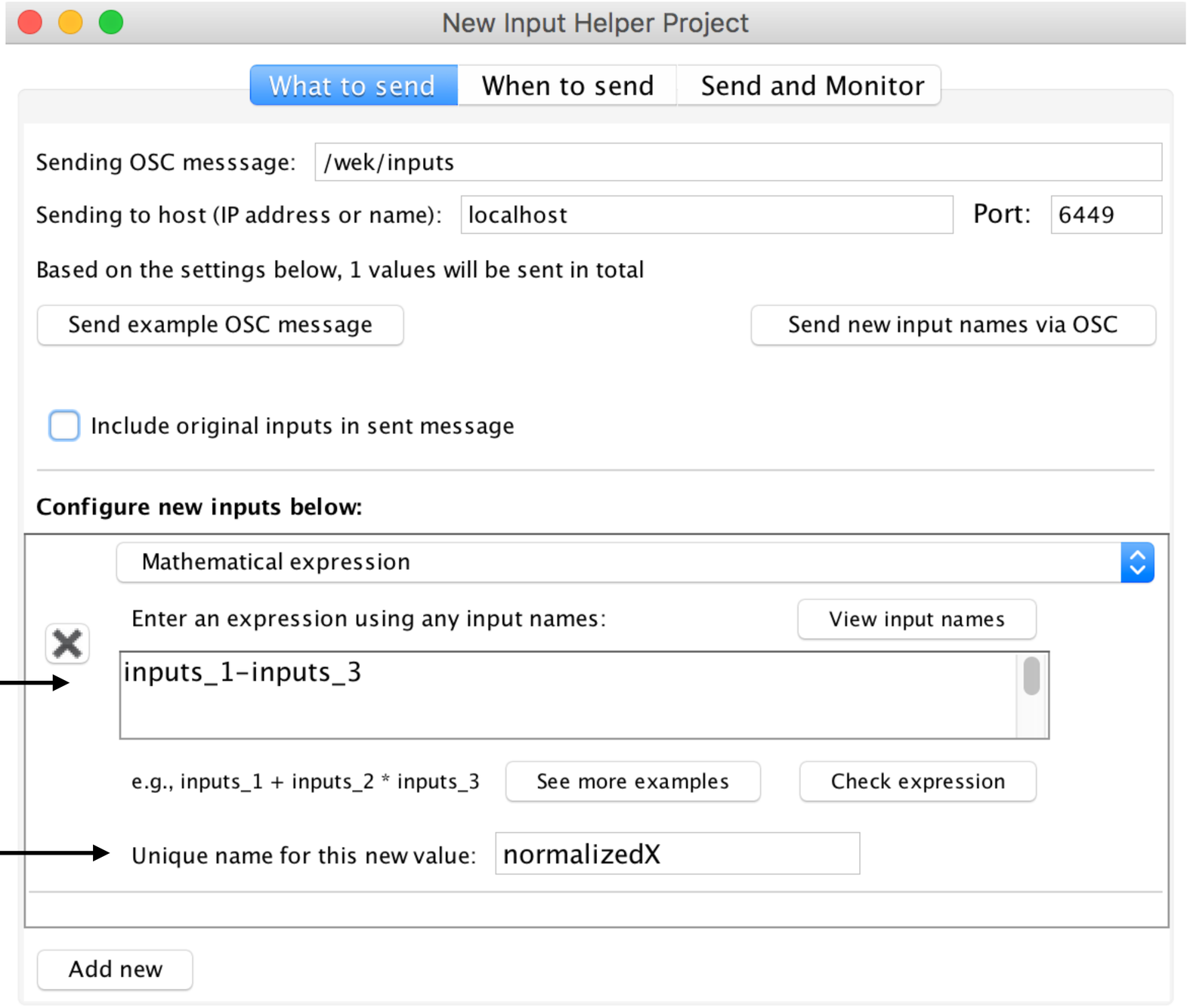
Click Add new to start ⟶

Scroll down and choose Mathematical expression

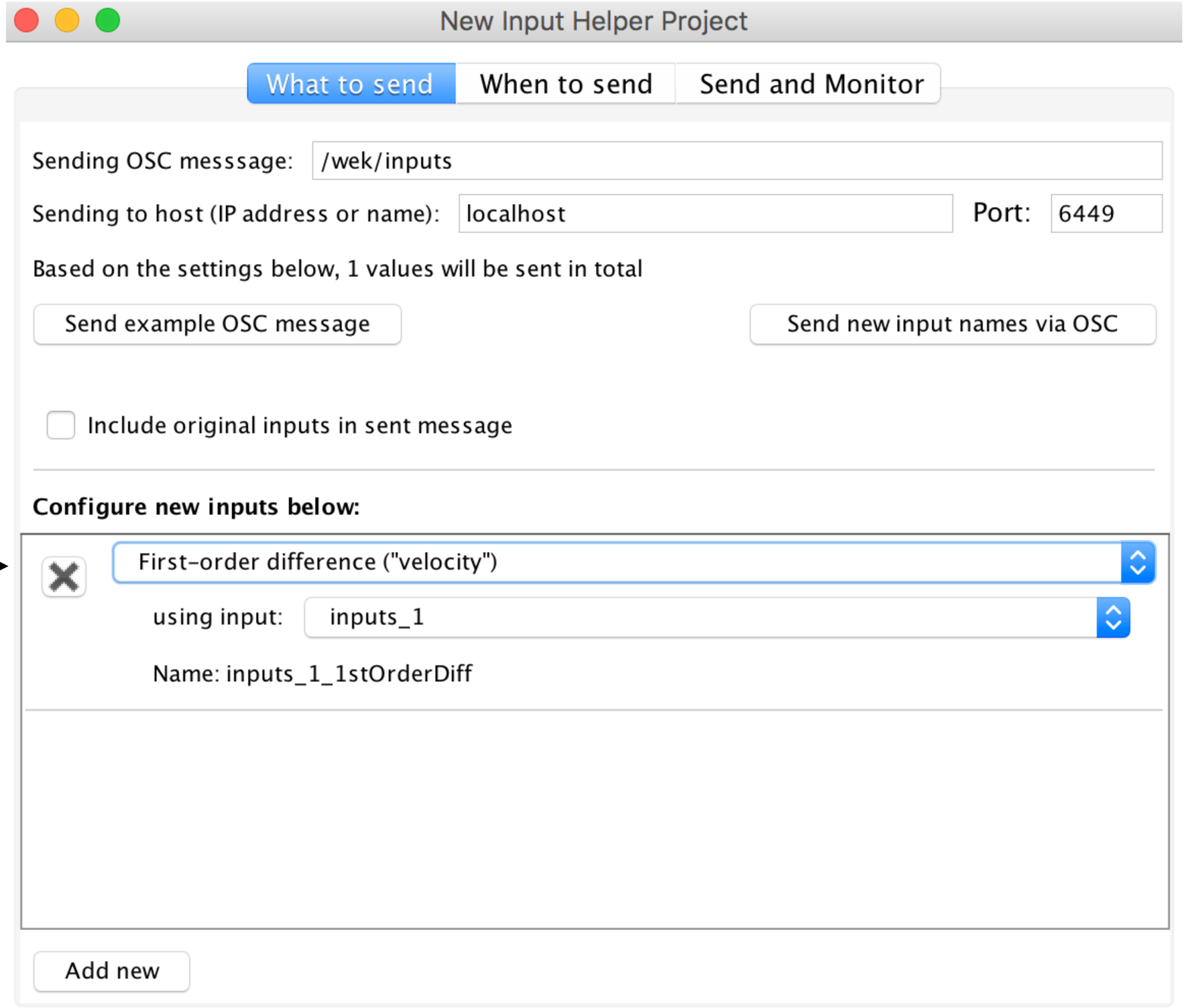


Enter in your equation

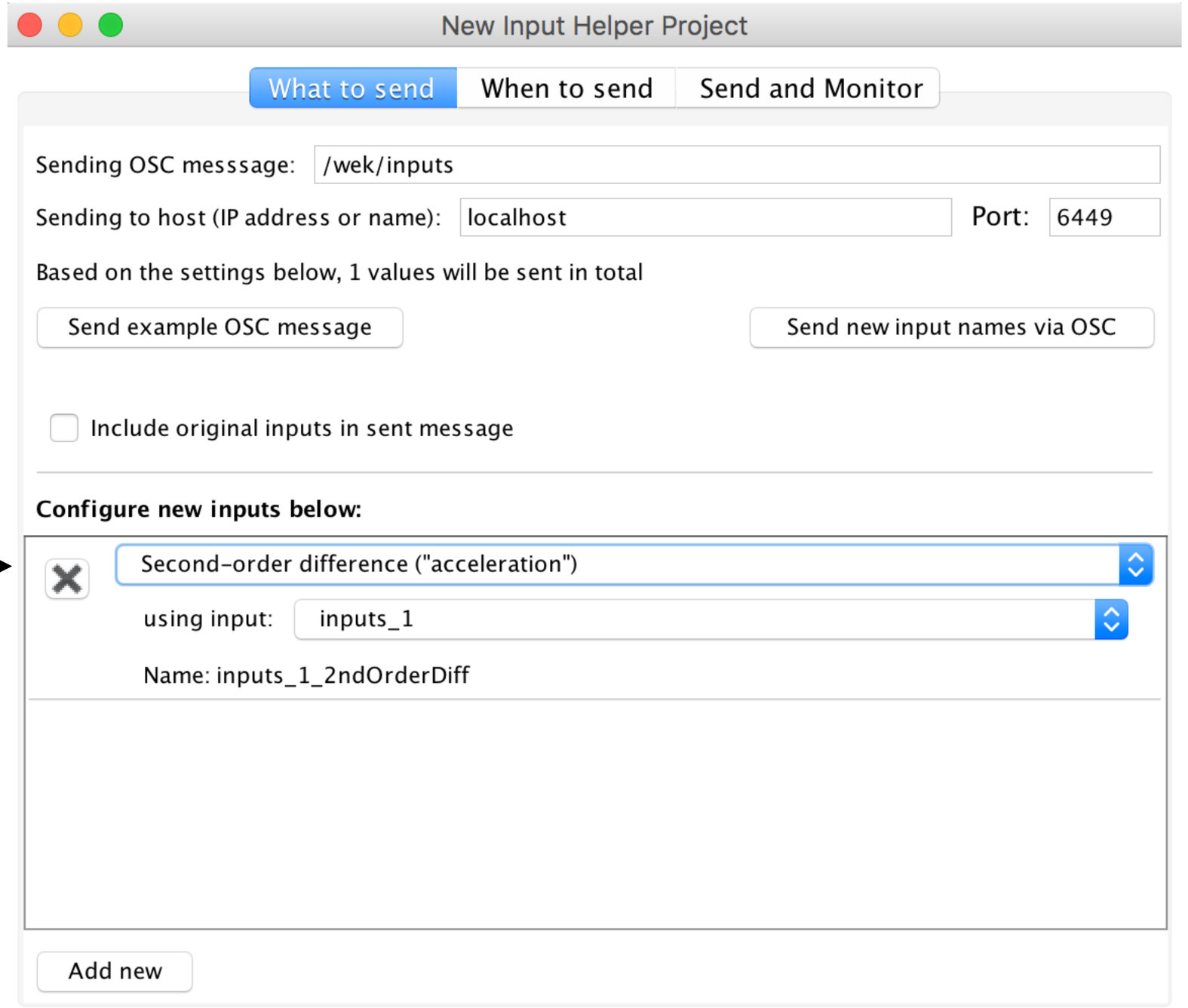
You can also custom name the output,which is helpful



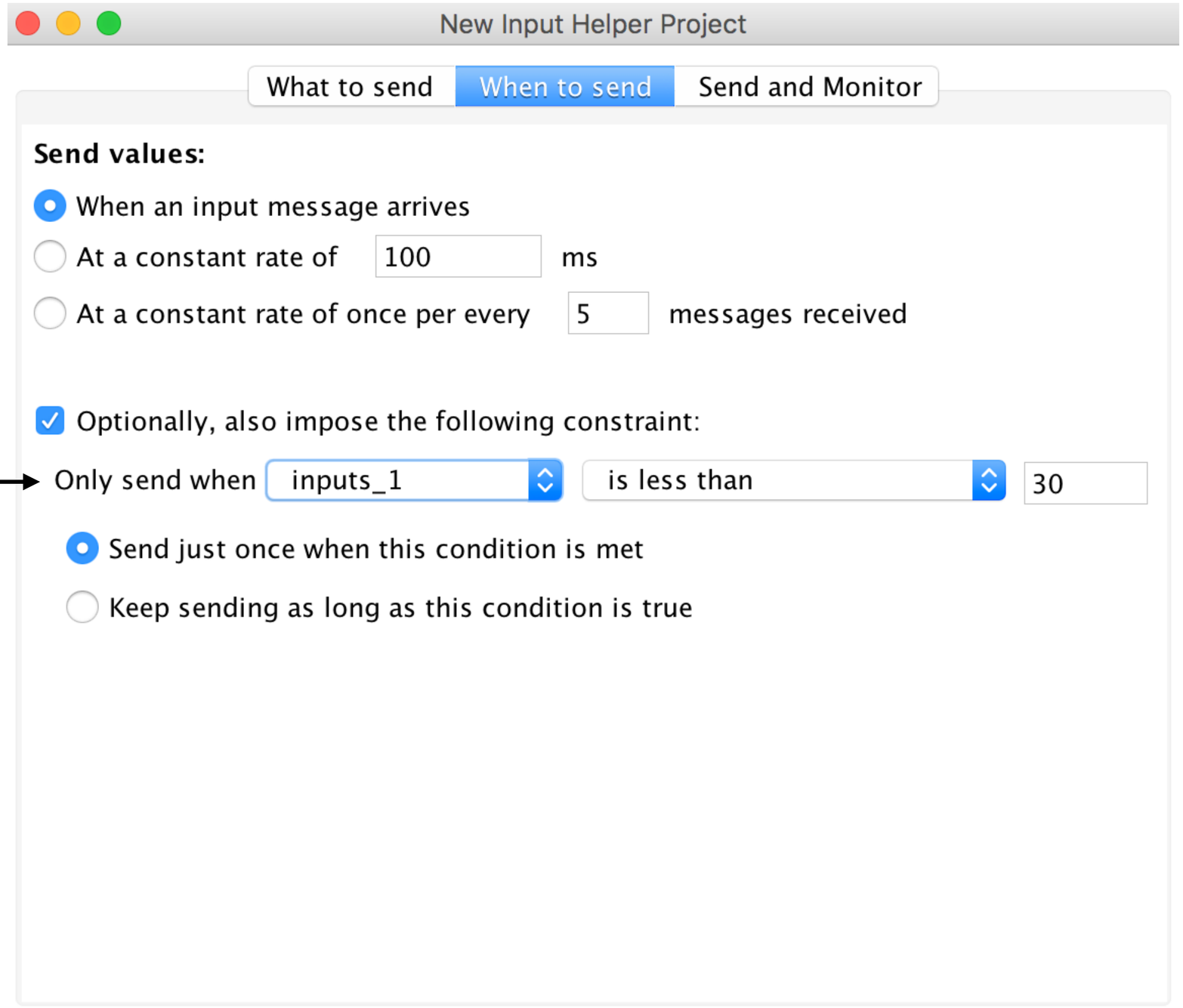
First-order difference is velocity, which could be though of as speed or force



Second-order difference is acceleration, which is a first order difference of velocity. It is actually the change in velocity and can be used to detect ‘strikes’ in input data.



If trying to detect a strike,
look at your data
and find a number
that is about where
the strike happens.
Then impose a constrain
that only once our data
goes below that number
then output OSC only once.



Audio as input

Various audio input analysis:

http://www.wekinator.org/examples/#Audio

- RMS - Root Mean Square
    - Good for volume

$$x_{\text{rms}} = \sqrt{\frac{1}{n}\left(x_1^2 + x_2^2 + \cdots + x_n^2\right)}$$

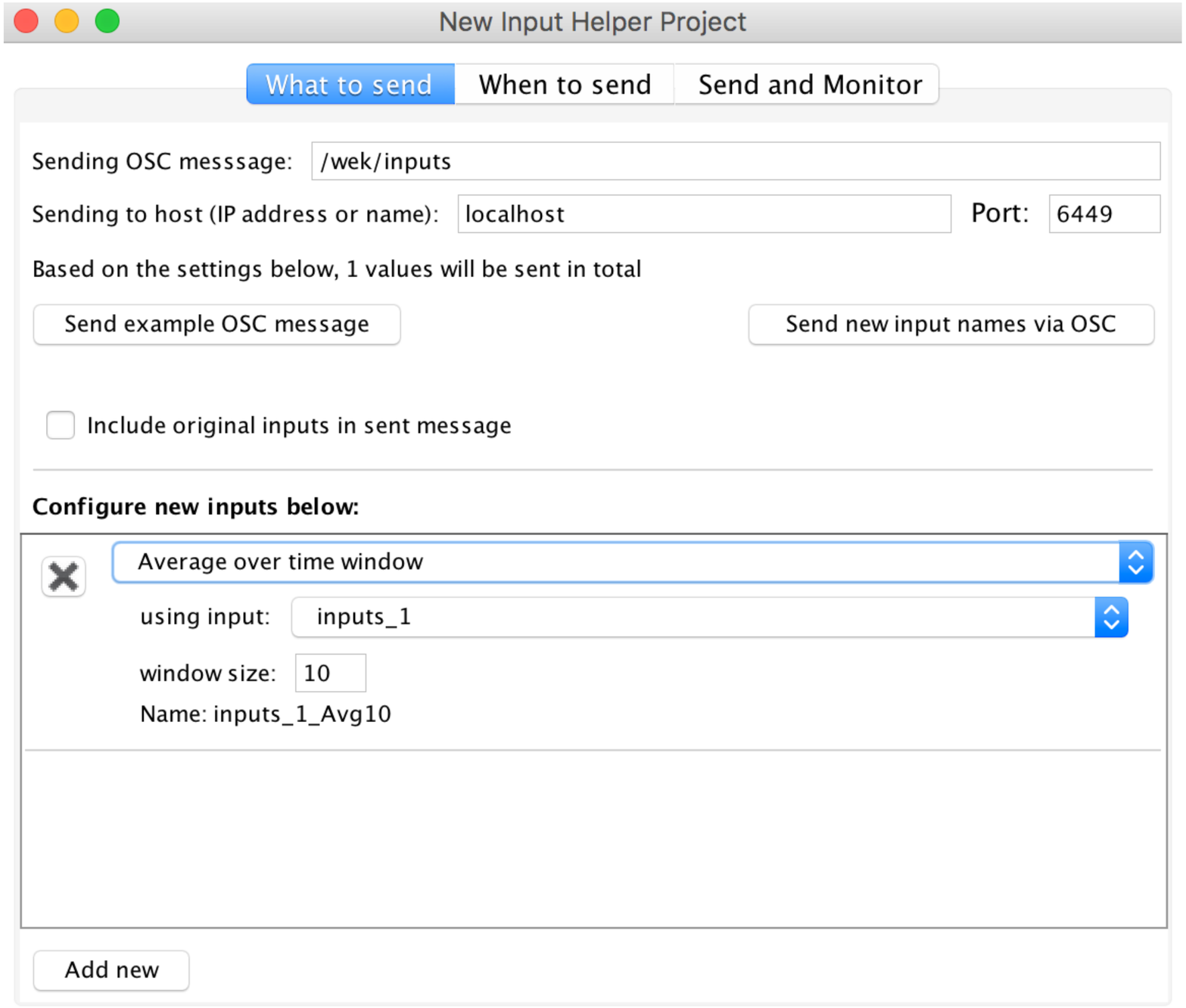
- FFT - Fast Fourier Transform
    - with constrained input sounds could build pitch or timbre classifier
- Peak Frequency (FFT)
    - could give us pitch of a note, or whether voice is male or female
- Spectral Centroid
    - will give us relative brightness of sound, warmer is lower, brighter is higher
    - could give us timbre
- Constant Q Transform
    - Same as FFT but with logarithmic space in-between bins, which is actually how we hear notes
- Mel Frequency Cepstral Coefficients (MFCCs)
    - Swiss army knife, pretty good for anything our than pitch

Smoothing

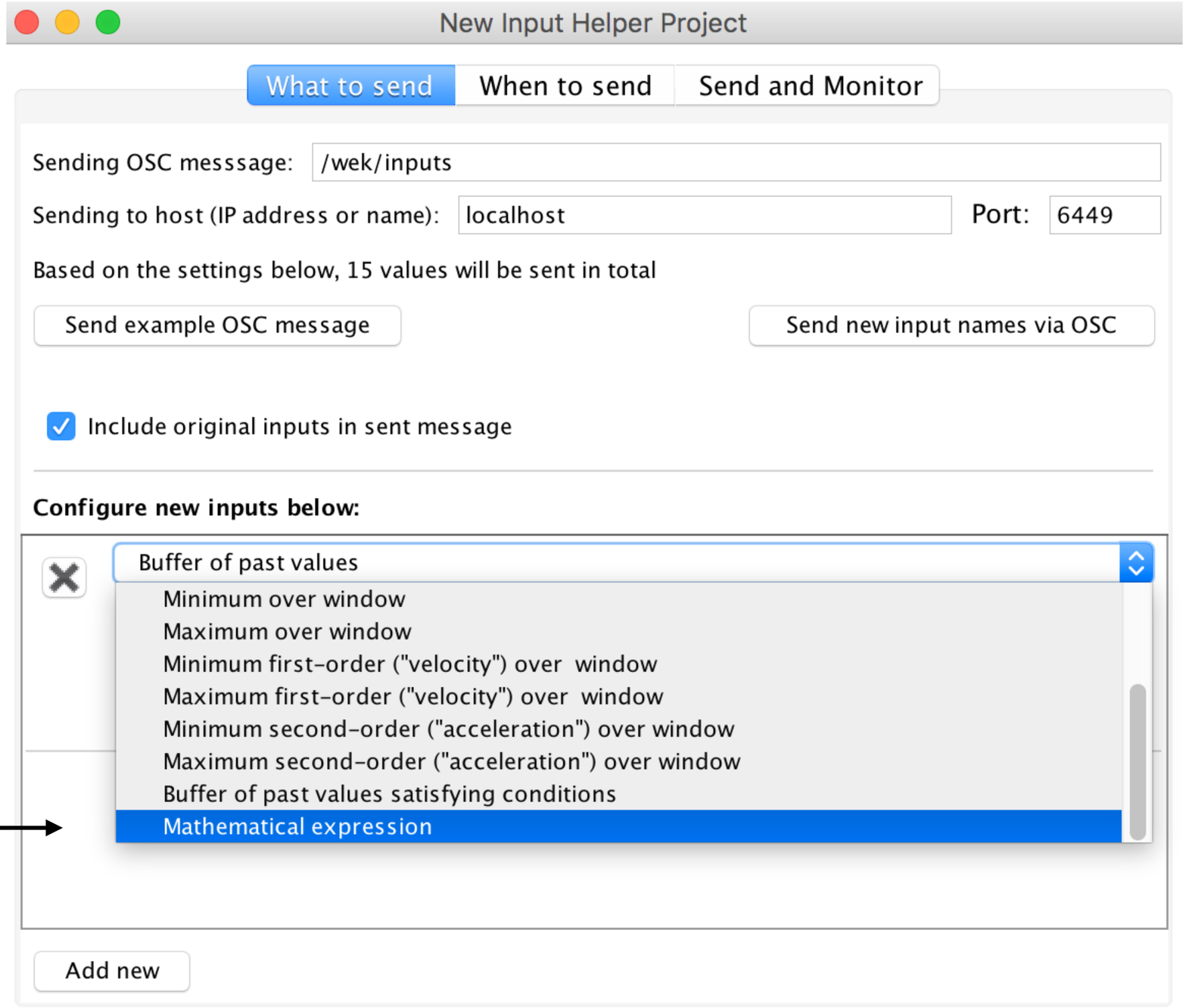
Get the distance from our current location (smoothed value) to the destination (the input value), scale it down, then add that to out current value:

smoothedValue += (inputValue - smoothedValue) * 0.1

You can take an
average over time
to smooth noisy signals.



You could also enter some signal processing equation as an expression.



Video

https://kylemcdonald.github.io/cv-examples/

https://github.com/kylemcdonald/ofxCv

https://github.com/moostrik/ofxFlowTools

Training Tips

- Usually more examples better
- But too much of same doesn't help
- Maybe send features at slow rate if building real time system, so features measurements are different