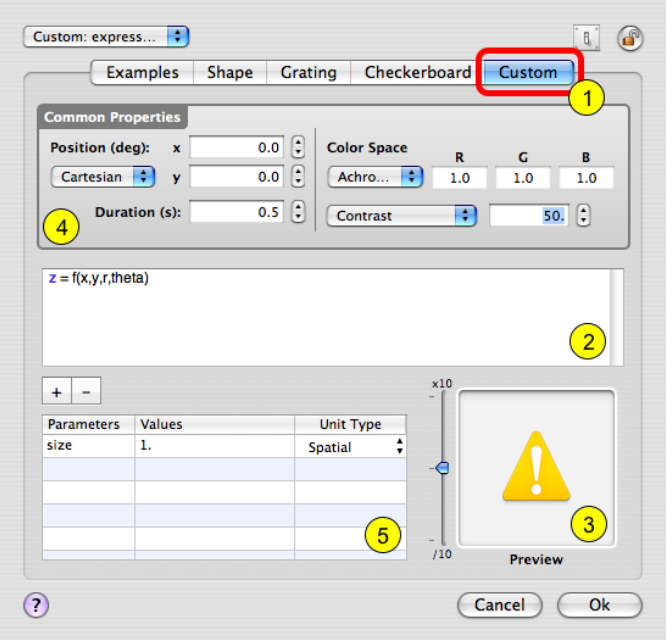
# Creating Filtered Noise Stimuli

This **Psykinematix** tutorial shows how to create low-pass and band-pass filtered 2D noise stimuli and combine them through masking using the expression evaluation capabilities of the "**Custom Stimulus**" panel.

## "Custom Stimulus" Panel




First, create a "**Visual Stimulus**" event in the "**Designer**" panel. Select this event and click on the properties button (or Apple-i). The panel above will open. Click on the "**Custom**" tab to select the "Custom Stimulus" type (1). From there, you can enter mathematical expressions in the text field (2). If correctly evaluated, a stimulus preview is displayed in the Preview box (3). The chromatic appearance of the stimulus is defined by the settings in Common Properties (4).

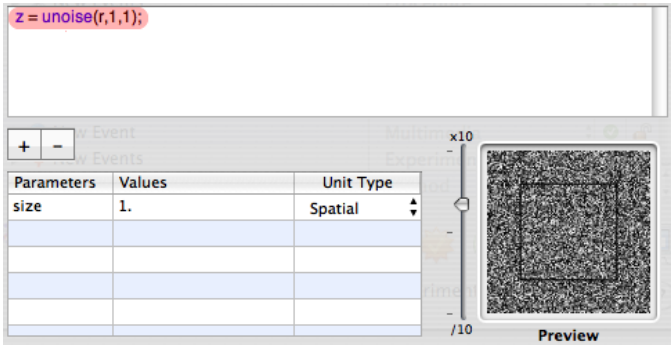
The expressions can also contain the parameters entered in the table (5); these parameters can either be associated with the dependent or independent variables of your experimental design. A *size* parameter is always predefined, so you can specify the width and height of the resulting 2D square array in degree units.

## Mathematical Expressions

The description of custom stimuli may consist of one-line expressions like '**z** = f(**x**, **y**, **r**, **theta**)' where **z** is the output in the [-1,1] range, (**x**, **y**) the Cartesian coordinates and (**r**, **theta**) the polar coordinates. **x**, **y**, **r**, and **theta** are built-in 2D parameters that specify the coordinates of each pixel. The stimulus description may also consist of multiple expressions split over several lines or separated by a semicolon.

## Uniform 2D Noise

First, let's create a 2D uniform white noise stimulus using the function ***unoise*** *(array size,seed,granularity)* where *array size* can be specified by either the **x**, **y**, **r** or **theta** coordinate arrays, *seed* is an integer specifying the seed to be used in the random generation (this way you can create correlated noise when using the same seed), and *granularity* is the size in pixels of a single dot.

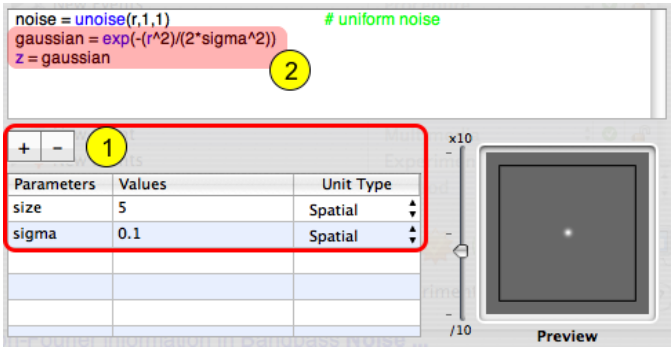



Note that the 2D uniform noise has an amplitude in the [-1, 1] range, and that the black square frame in the stimulus preview shows the stimulus boundary defined by the size parameter.

## Gaussian Filter

Now let's create the 2D Gaussian stimulus to filter the noise:

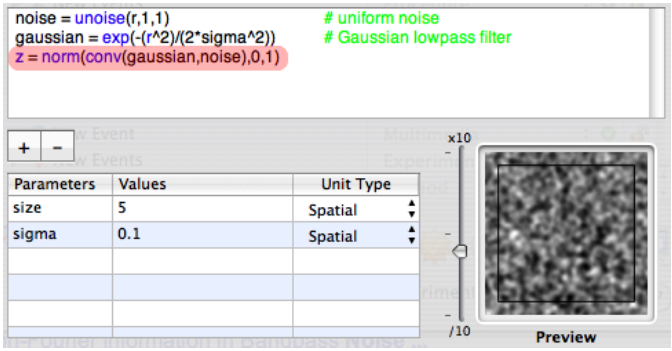
1) Add a *sigma* parameter to the table, set its value to 0.1 and its unit type to spatial (ie: 0.1 deg). Change the size value to 5 deg.
2) Add the expression for the Gaussian filter and assign it to the output variable **z**.




Note that we have renamed the noise component as an intermediary variable called '*noise*' but not used it yet, and we have added comments after the symbol # (shown in green).

## Isotropic Low-Pass Filtered Noise

Now that we have created the noise stimulus and the low-pass filter we wish to apply to it, let's convolve the two using the ***conv*** *(f, g)* function. Because the output has to be limited to the [-1, 1] range and the applied filter may not be normalized in terms of energy, a normalization is required to preserve the contrast of the filtered signal. This is done through the ***norm*** *(signal, mean, amplitude)* function that rescales the signal with the specified *mean* and maximum *amplitude*.

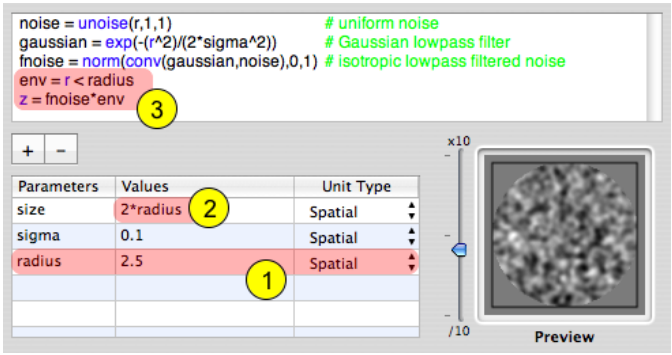



Note that the convolution function is performed using a highly optimized fast Fourier transform (FFT) algorithm.

## Circular Envelope

Finally, for proper stimulus presentation, the extent of the stimulus needs to be limited to a circular window. For convenience, let's add a *radius* parameter related to the size parameter:

1) Set the *radius* value to 2.5 deg (don't forget to set the unit type to *Spatial*).
2) Change the value of the *size* parameter to twice the radius (yes, you can also use any expression here!) so the size will scale automatically whenever the *radius* parameter changes.




To create the expression for the circular envelope as a hard-edge window (3), simply select all the pixels of the filtered noise whose radial position **r** is less than the window radius (ie: **r** < *radius*).

**Tip**: *Psykinematix understands the following comparison and logical operators:* < , > , & , | .

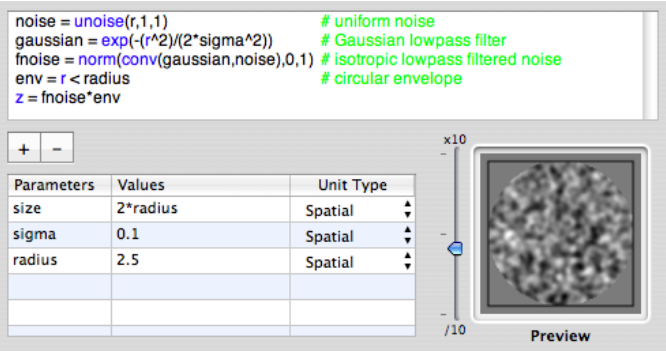
A KyberVision Innovation

Ψkinematix

Visual Psychophysics Made Easy !

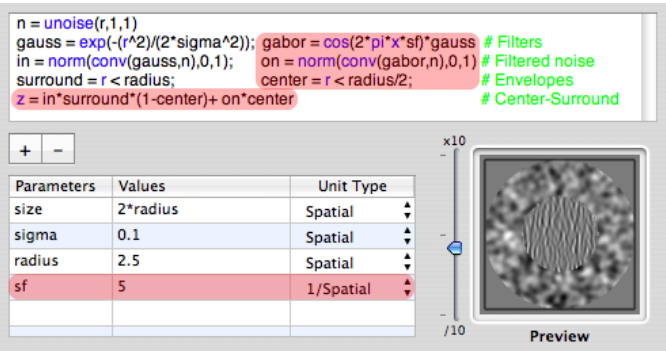
## It's never been easier to create complex 2D stimuli!!!

We have created isotropic low-pass filtered 2D white noise in just 5 lines! Unlike a Matlab solution there is no code overhead, and Psykinematix' expression evaluation is highly optimized for the PowerPC and Intel processors running Mac OS X.




## Not complex enough???

Adding a central circular area with band-pass filtered oriented noise is not much more difficult since it basically follows the same technique:

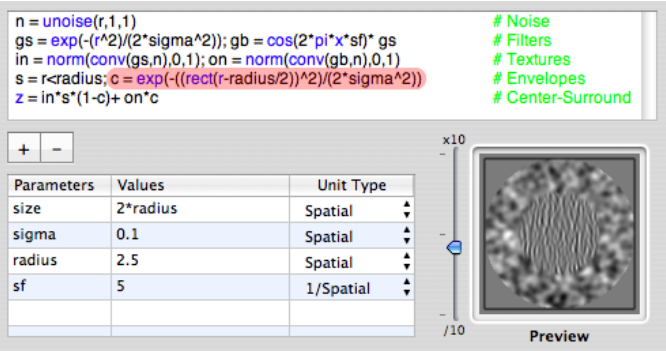



1) Create a band-pass filter named *gabor* (note the addition of a spatial frequency parameter *sf* in the table)
2) Convolve the initial noise with this oriented filter and name the result *on*
3) Create a mask for the central circular area and name it *center*
4) Combine the center and surround so the isotropic low-pass noise gets removed from the center (see the '1-*center* ' term) and replaced with the band-pass orientation noise

Still only 5 lines!

## Adding some polish...

The center-surround combination can be further improved by removing the hard edge transition between the two types of textures. This can be done by applying a small radial Gaussian modulation (of extent *sigma*) at the boundary of the center envelope (see variable *c*) to smooth the transition between the high and low frequency noise textures.




To perform this operation, compute the distance to the inner boundary for each pixel beyond it, to which is applied the Gaussian modulation: **rect** (**r**-*radius*/2) returns this distance if **r** >= *radius*/2, and 0 otherwise.

## See the Difference?

This stimulus can easily be integrated into any lessons in the **Orientation Discrimination Tutorial**!

We hope you have enjoyed this tutorial! If you have an idea for another one, let us know!!!