|  |  |  |  |
| --- | --- | --- | --- |
| 1 | function[ exp\_obj ] = q\_RSM( exp\_obj, stimulus ) | 1 | function[ exp\_obj ] = q\_RSM( exp\_obj, stimulus ) |
| 2 | % q\_RSM: This checks for some properties of stimulus. Mostly a switch to | 2 | % q\_RSM: This checks for some properties of stimulus. Mostly a switch to |
| 3 | % invoke the proper stimulus object constructor. Note that the returned | 3 | % invoke the proper stimulus object constructor. Note that the returned |
| 4 | % objects are placed in the queue of pending stimulus to be run. | 4 | % objects are placed in the queue of pending stimulus to be run. |
| 5 | % | 5 | % |
| 6 | % $Id: NAME VER\_ID DATA-TIME vinje $ | 6 | % $Id: NAME VER\_ID DATA-TIME vinje $ |
| 7 | % usage: NAME(Args) | 7 | % usage: NAME(Args) |
| 8 | % by: william vinje | 8 | % by: william vinje |
| 9 | % date: Date | 9 | % date: Date |
| 10 | % copyright: (c) Date William Vinje, Eduardo Jose Chichilnisky (GPL see RSM/COPYING) | 10 | % copyright: (c) Date William Vinje, Eduardo Jose Chichilnisky (GPL see RSM/COPYING) |
| 11 | % | 11 | % |
| 12 | [num\_pending, first\_nonempty] = Num\_Nonempty( exp\_obj.pending\_stimuli ); | 12 | [num\_pending, first\_nonempty] = Num\_Nonempty( exp\_obj.pending\_stimuli ); |
| 13 |  | 13 |  |
| 14 |  | 14 |  |
| 15 | % Check for valid exp\_obj | 15 | % Check for valid exp\_obj |
| 16 | if ( ~exist('exp\_obj') ) | 16 | if ( ~exist('exp\_obj') ) |
| 17 | fprintf('\t RSM ERROR: No experimental session object is present. Please run "Start\_RSM". \n'); | 17 | fprintf('\t RSM ERROR: No experimental session object is present. Please run "Start\_RSM". \n'); |
| 18 | return | 18 | return |
| 19 | end | 19 | end |
| 20 |  | 20 |  |
| 21 |  | 21 |  |
| 22 |  | 22 |  |
| 23 | % Check for valid stim class variable | 23 | % Check for valid stim class variable |
| 24 | if ( ~isfield(stimulus,'type') ) | 24 | if ( ~isfield(stimulus,'type') ) |
| 25 | fprintf('\t RSM ERROR: No valid stim class ("type") variable is present. Please assign stim class variable and try again. \n'); | 25 | fprintf('\t RSM ERROR: No valid stim class ("type") variable is present. Please assign stim class variable and try again. \n'); |
| 26 | return | 26 | return |
| 27 | end | 27 | end |
| 28 |  | 28 |  |
| 29 |  | 29 |  |
| 30 |  | 30 |  |
| 31 | if ( ~isfield(stimulus,'back\_rgb') ) | 31 | if ( ~isfield(stimulus,'back\_rgb') ) |
| 32 | fprintf('\t RSM WARNING: No valid background color variable is present. Setting background color to monitor default. \n'); | 32 | fprintf('\t RSM WARNING: No valid background color variable is present. Setting background color to monitor default. \n'); |
| 33 | stimulus.back\_rgb = exp\_obj.monitor.backgrndcolor; | 33 | stimulus.back\_rgb = exp\_obj.monitor.backgrndcolor; |
| 34 | end | 34 | end |
| 35 |  | 35 |  |
| 36 |  | 36 |  |
| 37 |  | 37 |  |
| 38 | if (~isfield(stimulus,'wait\_trigger')) | 38 | if (~isfield(stimulus,'wait\_trigger')) |
| 39 | stimulus.wait\_trigger = 0; | 39 | stimulus.wait\_trigger = 0; |
| 40 | end | 40 | end |
| 41 |  | 41 |  |
| 42 |  | 42 |  |
| 43 |  | 43 |  |
| 44 | if (~isfield(stimulus,'wait\_key')) | 44 | if (~isfield(stimulus,'wait\_key')) |
| 45 | stimulus.wait\_key = 0; | 45 | stimulus.wait\_key = 0; |
| 46 | end | 46 | end |
| 47 |  | 47 |  |
| 48 |  | 48 |  |
| 49 | if (isfield(stimulus,'trigger\_interval')) | 49 | if (isfield(stimulus,'trigger\_interval')) |
| 50 |  | 50 |  |
| 51 | exp\_obj.dio\_config.numframes\_per\_pulse = stimulus.trigger\_interval; | 51 | exp\_obj.dio\_config.numframes\_per\_pulse = stimulus.trigger\_interval; |
| 52 |  | 52 |  |
| 53 | end | 53 | end |
| 54 |  | 54 |  |
| 55 |  | 55 |  |
| 56 |  | 56 |  |
| 57 | switch stimulus.type | 57 | switch stimulus.type |
| 58 |  | 58 |  |
| 59 | case 'FS', % focus squares | 59 | case 'FS', % focus squares |
| 60 | exp\_obj.pending\_stimuli{num\_pending + 1} = Focus\_Squares(stimulus, exp\_obj); | 60 | exp\_obj.pending\_stimuli{num\_pending + 1} = Focus\_Squares(stimulus, exp\_obj); |
| 61 |  | 61 |  |
| 62 |  | 62 |  |
| 63 |  | 63 |  |
| 64 | case 'SC', % solid color | 64 | case 'SC', % solid color |
| 65 | stimulus.control\_flag = 4; | 65 | stimulus.control\_flag = 4; |
| 66 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo(stimulus, exp\_obj); | 66 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo(stimulus, exp\_obj); |
| 67 |  | 67 |  |
| 68 |  | 68 |  |
| 69 | case 'FC', % flashing color | 69 | case 'FC', % flashing color |
| 70 | stimulus.control\_flag = 3; | 70 | stimulus.control\_flag = 3; |
| 71 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo(stimulus, exp\_obj); | 71 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo(stimulus, exp\_obj); |
| 72 |  | 72 |  |
|  |  | 73 | case 'FP', % full-field pulses |
|  |  | 74 | stimulus.control\_flag = 5; |
|  |  | 75 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo(stimulus, exp\_obj); |
|  |  | 76 |  |
| 73 |  | 77 |  |
| 74 | case 'MB', | 78 | case 'MB', |
| 75 |  | 79 |  |
| 76 | exp\_obj.pending\_stimuli{num\_pending + 1} = Moving\_Bar(stimulus); | 80 | [stim, seq, trial\_num\_total] = rand\_stim(stimulus); |
|  |  | 81 | stim\_out = stimulus; |
|  |  | 82 | stim\_out.trial\_list = seq; |
|  |  | 83 | stim\_out.trials = stim; |
|  |  | 84 | uisave('stim\_out') |
|  |  | 85 |  |
|  |  | 86 |  |
|  |  | 87 | for i = 1:trial\_num\_total |
|  |  | 88 | exp\_obj.pending\_stimuli{num\_pending + i} = Moving\_Bar(stim(i)); |
|  |  | 89 | end |
|  |  | 90 |  |
|  |  | 91 | duration = calc\_mb\_duration(exp\_obj); |
|  |  | 92 | fprintf('STIMULUS DURATION: %d Seconds\n', duration) |
| 77 |  | 93 |  |
| 78 |  | 94 |  |
| 79 | case 'MG', | 95 | case 'MG', |
| 80 | exp\_obj.pending\_stimuli{num\_pending + 1} = Moving\_Grating(stimulus, exp\_obj); | 96 | [stim, seq, trial\_num\_total] = rand\_stim(stimulus); |
|  |  | 97 | stim\_out = stimulus; |
|  |  | 98 | stim\_out.trial\_list = seq; |
|  |  | 99 | stim\_out.trials = stim; |
|  |  | 100 | uisave('stim\_out') |
|  |  | 101 |  |
|  |  | 102 |  |
|  |  | 103 |  |
|  |  | 104 | for i = 1:trial\_num\_total |
|  |  | 105 | exp\_obj.pending\_stimuli{num\_pending + i} = Moving\_Grating(stim(i), exp\_obj); |
|  |  | 106 | end |
| 81 |  | 107 |  |
| 82 | mglSetParam('visualAngleSquarePixels',0,1); | 108 | mglSetParam('visualAngleSquarePixels',0,1); |
| 83 | mglVisualAngleCoordinates(exp\_obj.rig\_geom.optical\_path\_length,[exp\_obj.monitor.physical\_width, exp\_obj.monitor.physical\_height]); | 109 | mglVisualAngleCoordinates(exp\_obj.rig\_geom.optical\_path\_length,[exp\_obj.monitor.physical\_width, exp\_obj.monitor.physical\_height]); |
| 84 |  | 110 |  |
| 85 |  | 111 |  |
| 86 | case 'CG', | 112 | case 'CG', |
| 87 | exp\_obj.pending\_stimuli{num\_pending + 1} = Counterphase\_Grating(stimulus, exp\_obj); | 113 | [stim, seq, trial\_num\_total] = rand\_stim(stimulus); |
|  |  | 114 | stim\_out = stimulus; |
|  |  | 115 | stim\_out.trial\_list = seq; |
|  |  | 116 | stim\_out.trials = stim; |
|  |  | 117 | uisave('stim\_out') |
|  |  | 118 |  |
|  |  | 119 | for i = 1:trial\_num\_total |
|  |  | 120 | exp\_obj.pending\_stimuli{num\_pending + i} = Counterphase\_Grating(stim(i), exp\_obj); |
|  |  | 121 | end |
| 88 |  | 122 |  |
| 89 | mglSetParam('visualAngleSquarePixels',0,1); | 123 | mglSetParam('visualAngleSquarePixels',0,1); |
| 90 | mglVisualAngleCoordinates(exp\_obj.rig\_geom.optical\_path\_length,[exp\_obj.monitor.physical\_width, exp\_obj.monitor.physical\_height]); | 124 | mglVisualAngleCoordinates(exp\_obj.rig\_geom.optical\_path\_length,[exp\_obj.monitor.physical\_width, exp\_obj.monitor.physical\_height]); |
| 91 |  | 125 |  |
| 92 |  | 126 |  |
| 93 | case 'RN', % random and noise | 127 | case 'RN', % random and noise |
| 94 | exp\_obj.pending\_stimuli{num\_pending + 1} = Random\_Noise\_Binary\_LUT(stimulus, exp\_obj); | 128 | exp\_obj.pending\_stimuli{num\_pending + 1} = Random\_Noise\_Binary\_LUT(stimulus, exp\_obj); |
| 95 |  | 129 |  |
| 96 | case 'RG', % random and noise | 130 | case 'RG', % random and noise |
| 97 | exp\_obj.pending\_stimuli{num\_pending + 1} = Random\_Noise\_CDF\_LUT(stimulus, exp\_obj); | 131 | exp\_obj.pending\_stimuli{num\_pending + 1} = Random\_Noise\_CDF\_LUT(stimulus, exp\_obj); |
| 98 |  | 132 |  |
| 99 |  | 133 |  |
| 100 | case 'RM', % raw movie | 134 | case 'RM', % raw movie |
| 101 | exp\_obj.pending\_stimuli{num\_pending + 1} = Raw\_Movie( stimulus, exp\_obj ); | 135 | exp\_obj.pending\_stimuli{num\_pending + 1} = Raw\_Movie( stimulus, exp\_obj ); |
| 102 |  | 136 |  |
| 103 |  | 137 |  |
| 104 | case 'PL', % pulse for cone isolation | 138 | case 'PL', % pulse for cone isolation |
| 105 | stimulus.control\_flag = 1; | 139 | stimulus.control\_flag = 1; |
| 106 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo( stimulus, exp\_obj ); | 140 | exp\_obj.pending\_stimuli{num\_pending + 1} = PulseCombo( stimulus, exp\_obj ); |
| 107 |  | 141 |  |
| 108 |  | 142 |  |
| 109 | otherwise, | 143 | otherwise, |
| 110 | fprintf('\t RSM ERROR: Stim class variable not recognized. Please assign different stim class variable and try again. \n'); | 144 | fprintf('\t RSM ERROR: Stim class variable not recognized. Please assign different stim class variable and try again. \n'); |
| 111 | return | 145 | return |
| 112 |  | 146 |  |
| 113 | end % switch | 147 | end % switch |
|  |  | 148 |  |
|  |  | 149 | if (stimulus.wait\_trigger) |
|  |  | 150 |  |
|  |  | 151 | % Wait for main trigger signal from DAQ |
|  |  | 152 | fprintf('WAITING FOR TRIGGER: %s \n', stimulus.type); |
|  |  | 153 | Scan\_4\_Trigger( exp\_obj ); % recall the timestamp occurs within Scan\_4\_Trigger |
|  |  | 154 |  |
|  |  | 155 | elseif (stimulus.wait\_key) |
|  |  | 156 | fprintf('WAITING FOR KEY: %s \n', stimulus.type); |
|  |  | 157 | % wait around and check for trigger event |
|  |  | 158 | pause; % wait for key press event |
|  |  | 159 |  |
|  |  | 160 | end % wait for main trigger event |
| 114 |  | 161 |  |