MPM>find_pitch (142 calls, 612.752 sec)

Generated 16-Mar-2015 23:25:46 using real time.
subfunction in file /Users/zichaowang/Dropbox/RICE/year spring 15/ELEC 490/pitch detection/my workspace/MPM.m
Copy to new window for comparing multiple runs

Refresh

Show parent functions Show busy lines Show child functions
Show Code Analyzer results Show file coverage Show function listing

Parents (calling functions)

Function Name	Function Type	Calls
MPM	function	142

Lines where the most time was spent

Line Number	Code	Calls	Total Time	% Time	Time Plot
69	`m_tau(tau+1) = m_tau(tau+1) + ...`	297940992	170.659 s	27.9%	
70	`n_tau(tau+1) = 2*r_tau(tau+1)/...`	297940992	155.099 s	25.3%	
68	`r_tau(tau+1) = r_tau(tau+1) + ...`	297940992	154.078 s	25.1%	
71	`end`	297940992	130.471 s	21.3%	
92	`max_idx = find(n_tau==temp_max...`	148314	0.743 s	0.1%	
All other lines			1.702 s	0.3%	
Totals			612.752 s	100%	

Children (called functions)
No children

Code Analyzer results

Line number	Message
78	`The value assigned to variable 'max_idx' might be unused.`
93	`If 'local_max' is an indexed variable, performance can be improved using logical indexing instead of FIND.`

Coverage results
Show coverage for parent directory

Total lines in function	49
Non-code lines (comments, blank lines)	6
Code lines (lines that can run)	43

ılate ACF
%calculate SDF
late NSDF

1)

, which is 1
d later
ve element

Code lines that did run	36
Code lines that did not run	7
Coverage (did run/can run)	83.72 %

Function listing

Color highlight code according to time

time	calls	line	
		61	function [f, note] = find_pitch(x,W,fs)
< 0.01	142	62	r_tau = zeros(1,W); % initialize ACF
	142	63	m_tau = zeros(1,W); % initialize SDF
	142	64	n_tau = zeros(1,W); % initialize NSDF
		65	%% calculate NSDF (normalized square difference function)
< 0.01	142	66	for tau = 0:W-1
0.22	290816	67	for j = 1:1+W-tau-1
154.08	297940992	68	r_tau(tau+1) = r_tau(tau+1) + x(j)*x(j+tau); % calcu
170.66	297940992	69	m_tau(tau+1) = m_tau(tau+1) + (x(j)^2+ x(j+tau)^2);%
155.10	297940992	70	n_tau(tau+1) = 2*r_tau(tau+1)/m_tau(tau+1); % calcu
130.47	297940992	71	end
0.63	290816	72	end
		73	%% find local maxima
< 0.01	142	74	MAX = max(n_tau); % maximum correlation in NSDF (usually 1
	142	75	k = 0.8; % threshold parameter
	142	76	th = MAX*k; % threshold for selecting key maximum
	142	77	idx = 2; % starting index, excluding the first data point,
	142	78	max_idx = 0; % the index of the key maximum. To be changed
< 0.01	142	79	temp = find(n_tau<0); % for finding index of first negativ
< 0.01	142	80	local_max = zeros(1,W); % local maximum in NSDF
< 0.01	142	81	while idx < length(n_tau)-1 % for each sample
0.08	148314	82	temp_max = 0;
0.06	148314	83	while n_tau(idx) > 0 && idx < W && idx > temp(1)
		84	% for sample starting from the second arising pattern
0.06	142260	85	if n_tau(idx+1) > n_tau(idx)
0.03	71040	86	if temp_max < n_tau(idx+1)
0.03	66628	87	temp_max = n_tau(idx+1);
0.02	66628	88	end
0.03	71040	89	end
0.06	142260	90	idx = idx + 1;
0.06	142260	91	end
0.74	148314	92	max_idx = find(n_tau==temp_max);
0.13	148314	93	local_max(max_idx) = temp_max;
0.09	148314	94	idx = idx + 1;
0.17	148314	95	end
		96	%% find pitch and fundamental frequency
	142	97	tau = find(local_max>th); % pitch period
	142	98	if ~isempty(tau) % check if there is a key maximum
< 0.01	142	99	f = fs / tau(1); % corresponding frequency

ɲvert to midi mappings

```
                  100          %note = log10(f/27.5)/log10(2^(1/12));
          142     101          note = round((12 * log10(f/440)/log10(2)) + 69); % Cor
                  102      else
                  103          f = -1;
                  104          note = -1;
                  105      end
< 0.01    142     106      if note < 0 % deal with noise / no sound
                  107          f = -1;
                  108          note = -1;
                  109      end
```

Other subfunctions in this file are not included in this listing.