RayTracer>RayTracer.findCurrentQuadrant (Calls: 23374568, Time: 625.637 s)

Generated 27-Mar-2023 12:54:44 using performance time.
Class method in file /Users/lhess/ffr-photon-simulator-matlab/mask-simulator/RayTracer.m
Copy to new window for comparing multiple runs

Parents (calling functions)

Lines that take the most time

Line Number	Code	Calls	Total Time (s)	% Time	Time Plot
60	`rightBounds = [quadrants.rightBound];`	23374568	160.852	25.7%	
83	`currentQuadrant = quadrants(rightBounds >= photon.x & …`	23374568	138.860	22.2%	
61	`leftBounds = [quadrants.leftBound];`	23374568	132.609	21.2%	
52	`quadrantLayer = quadrantLayers(quadrantLayers(i).outer…`	23374568	118.318	18.9%	
90	`end`	23374568	36.133	5.8%	
All other lines			38.864	6.2%	
Totals			625.637	100%	

Children (called functions)

No children

Code Analyzer results

Line Number	Message
82	`The value assigned to variable 'currentQuadrant' might be unused.`

Coverage results

Show coverage for parent folder

Total lines in function	51
Non-code lines (comments, blank lines)	42
Code lines (lines that can run)	9
Code lines that did run	9
Code lines that did not run	0
Coverage (did run/can run)	100.00 %

Function listing

```
Time      Calls     Line
                    40          function currentQuadrant = findCurrentQuadrant(obj, photon)
                    41            % Iterate through every quadrant in the current FFR layer and compare
                    42            % the photon's coordinates to the quadrant's boundaries to determine
                    43            % which quadrant the photon is currently in.
                    44            %Debug.msg('Finding current quadrant.', 1);
  8.877  23374568   45            quadrantLayers = obj.currFFRLayer.quadrantLayers;
                    46
                    47            % Vectorize the iteration through the quadrant layers.
 11.760  23374568   48            i = 1:obj.currFFRLayer.nQLayers;
                    49            % Use the photon's y coordinate to narrow down which QuadrantLayer it's in.
                    50            % The quadrantLayers list goes in increasing innerBound height order, so check
                    51            % if photon.y is less than the outerBound (not greater than). Use logical indexing.
118.318  23374568   52            quadrantLayer = quadrantLayers(quadrantLayers(i).outerBound > photon.y);
                    53
                    54            % Use logical indexing and vectorization to isolate the quadrant in the list which contains the photon.
                    55            % This means finding the common index between the left and right quadrant bounds which the photon can be
  4.476  23374568   56            quadrants = quadrantLayer.quadrants;
                    57
                    58            % Get the left and right bounds of each quadrant.
                    59            %i = 1:single(quadrantLayer.nQuadrants);
160.852  23374568   60            rightBounds = [quadrants.rightBound];
132.609  23374568   61            leftBounds = [quadrants.leftBound];
                    62
                    63            % Preallocate the rightBounds and leftBounds array to test for
```

```
                    64            % speed improvements.
                    65            %rightBounds = nan(single(quadrantLayer.nQuadrants));
                    66            %leftBounds  = nan(single(quadrantLayer.nQuadrants));
                    67            %rightBounds = nan(i);
                    68            %leftBounds = nan(i);
                    69
                    70            %currentQuadrant = [];
                    71            %for i = 1:single(quadrantLayer.nQuadrants)
                    72            %    quadrant = quadrants(i);
                    73            %    if quadrant.rightBound >= photon.x && quadrant.leftBound < photon.x
                    74            %        currentQuadrant = quadrant;
                    75            %        break;
                    76            %    end
                    77            %end
                    78
                    79
                    80            % Find the common index and get the value, i.e. quadrants([0 1 1] & [1 1 0]) -> quadrants([0 1 0])
                    81            % means the photon is in the middle quadrant.
  0.590 23374568    82            currentQuadrant = [];
138.860 23374568    83            currentQuadrant = quadrants(rightBounds >= photon.x & leftBounds < photon.x);
                    84
                    85            % Fail with a custom alert.
                    86            %if isempty(currentQuadrant)
                    87            %  Debug.alert('Current quadrant not found.', 0);
                    88            %  Debug.msg('Photon coords: ' + obj.coordToString([ photon.x photon.y ]), 0);
                    89            %end
 36.133 23374568    90        end
```

Local functions in this file are not included in this listing.