```
classdef KdNode < handle
    %KdNode node in a 2D K-d tree

    properties
        config    % Configuration this node represents
        left      % Left node of this node
        right     % Right node of this node
        split     % Integer 1-3 which config.th to split on
    end

    methods
        function obj = KdNode(config, left, right)
            %KdNode Construct an instance of this class
            obj.config = config;
            obj.left = left;
            obj.right = right;
        end

        function th = cval(obj, i)
            th = obj.config.th(i);
        end

        function insert(obj, node)
            % If split value of node less than current, go left
            if node.cval(obj.split) <= obj.cval(obj.split)
                % If no left node
                if isa(obj.left, 'NullKdNode')
                    % Increment split criterion
                    node.split = mod(obj.split+1, 4);
                    if node.split == 0
                        node.split = node.split + 1;
                    end
                    obj.left = node;
                else
                    obj.left.insert(node);
                end
            % Go right
            else
                % If no right node
                if isa(obj.right, 'NullKdNode')
                    % Increment split criterion
                    node.split = mod(obj.split+1, 4);
                    if node.split == 0
                        node.split = node.split + 1;
                    end
                    obj.right = node;
                else
                    obj.right.insert(node);
                end
            end
        end
```

```
        function near = nearest(obj, config, curr_near, curr_dist,
path)
            path.insert(obj.config);
            % If same as this node, return this node config
            if isequal(config.th, obj.config.th)
                near = obj.config;
            else
                dist = norm(obj.config.th - config.th);
                % If this node closer than current closest
                if dist < curr_dist
                    near = obj.config;
                    curr_near = obj.config;
                    curr_dist = dist;
                else
                    near = curr_near;
                end

                % If there's a left or a right
                if isa(obj.left, 'KdNode') || isa(obj.right, 'KdNode')
                    % If split value of node less than current, go
left
                    if config.th(obj.split) <= obj.cval(obj.split)
                        % If left node, recurse, otherwise do nothing
                        if isa(obj.left, 'KdNode')
                            near = obj.left.nearest(config, curr_near,
curr_dist, path);
                        end
                    % Go right
                    else
                        % If right node, recurse, otherwise do nothing
                        if isa(obj.right, 'KdNode')
                            near = obj.right.nearest(config,
curr_near, curr_dist, path);
                        end
                    end
                end
            end

            function draw(obj, color)
                % Draw this node's config
                obj.config.draw(color);
                if isa(obj.left, 'KdNode')
                    obj.left.draw(color);
                end
                if isa(obj.right, 'KdNode')
                    obj.right.draw(color);
                end
            end
        end
    end
```

Published with MATLAB® R2020a